Bachelorarbeit

Ein Evolutionärer Algorithmus zur Erstellung eines künstlichen Gegenspielers für „Chinesische Mauer“

Erarbeitet von: Thomas Arnold

Betreut von: Prof. Johannes Fürnkranz
Fachbereich: Informatik
TU Darmstadt

14.10.07

Hiermit versichere ich, dass ich die Arbeit selbständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.
Fürth, 14.10.07

Zusammenfassung

„Chinesische Mauer“ ist der Name eines Kartenspiels für zwei bis fünf Personen. Das Ziel dieser Bachelor-Arbeit ist die Entwicklung eines künstlichen Gegenspielers mit Hilfe eines evolutionären Algorithmus. Dieser Algorithmus basiert auf der biologischen Evolution und besitzt daher Mechanismen wie Mutation und Selektion. Dadurch wird der Gegenspieler immer weiter verbessert.

In dieser Arbeit wird zunächst das Kartenspiel „Chinesische Mauer“ vorgestellt.

Im nächsten Kapitel wird kurz der generelle Ansatz eines evolutionären Algorithmus erläutert und erklärt.

Danach wird die konkrete Umsetzung und Entwicklung des Algorithmus beschrieben, dabei wird auf entstandene Probleme und deren Lösungsansätze eingegangen. Es wird auch die Funktionsweise des einprogrammierten, künstlichen Gegenspielers erklärt.

Schließlich muss die Spielstärke des entwickelten Gegenspielers eingeschätzt und analysiert werden, was jedoch keine einfache Aufgabe ist. Dazu werden die entsprechenden Ergebnisse vorgestellt.

Abschließend wird das Ergebnis diskutiert: es wird besprochen, ob der Algorithmus ein gewünschtes Ergebnis liefern konnte, und es werden Verbesserungsmöglichkeiten aufgezeigt, die andere Ansätze beinhalten oder den Rahmen dieser Arbeit übersteigen.

Inhaltsverzeichnis

1 Das Spiel „Chinesische Mauer“

Dieses Kapitel stellt kurz das Spiel vor und erklärt dessen Regeln.

1.1 Vorstellung

„Chinesische Mauer“ ist ein Kartenspiel für zwei bis fünf Spieler ab 10 Jahren (Angabe des Herstellers). Es wurde im Jahr 2006 vom erfolgreichen deutschen Spieleautor Reiner Knizia entwickelt und in Deutschland über den Kosmos-Verlag verkauft. Ein Spiel dauert in etwa 30 Minuten, jedoch variiert die Spieldauer ein wenig mit der Anzahl der Spieler. Der Glücksfaktor ist moderat, man zieht beispielsweise Karten von einem gemischten Stapel. Jedoch besitzt der Spieler in seinem Zug sehr viele Möglichkeiten, wie er seinen Zug gestalten kann. Der Einsatz der richtigen Taktik ist generell entscheidender als das Kartenglück und macht den Reiz des Spieles aus. Immer wieder muss man zwischen aktivem Eingreifen an der richtigen Stelle und passivem Abwarten und dem Sammeln von Karten abwägen.

1.2 Regeln

In diesem Kapitel werden die Regeln des Spiels vorgestellt und anhand von Beispielen erklärt. Das Verständnis der Regeln wird in den folgenden Kapiteln angenommen und ist auch notwendig, wenn man den Ausführungen über die Vorgehensweisen des künstlichen Gegenspielers folgen möchte.

Das komplette Regelwerk ist übrigens zur Entstehungszeit dieser Arbeit frei auf der Internetseite des Verlags im pdf-Format verfügbar und kann auf der Webseite des deutschen Vertreibers KOSMOS heruntergeladen werden. (Siehe Quelle [7])

Aus diesem Dokument wurden die Inhalte für die Abbildungen in diesem Kapitel entnommen.

1.2.1 Komponenten

Es gibt zum einen die so genannten Ruhmeskärtchen. Dies sind kleine Marken, auf denen lediglich eine Zahl zwischen 1 und 8 abgebildet ist, wobei es kein Kärtchen mit dem Wert 6 gibt. Diese Zahl bestimmt die Anzahl an Ruhmespunkten, die das Ruhmeskärtchen wert ist.

Abbildung 1: Die verschiedenen Ruhmeskärtchen

Außerdem gibt es je 20 Mauerkarten in fünf verschiedenen Farben. Es gibt sieben verschiedene Mauerkarten, aber in jeder Farbe ist die Zusammenstellung der Mauerkarten gleich. Jeder Spieler entscheidet sich bei Spielbeginn für eine Farbe und hat so seinen eigenen Kartenstapel. In der linken, oberen Ecke jeder Mauerkarte ist mit einer Zahl die Stärke der Karte angegeben, diese kann jedoch durch diverse Spezialeffekte verändert werden. Die Besonderheiten der verschiedenen Mauerkarten werden in dem Abschnitt „Die einzelnen Karten und ihre Bedeutungen“ genauer erklärt.

Abbildung 2: Beispiele für Mauerkarten

1.2.2 Spielziel

Die Spieler sind Bauherren, die mit ihrem Einsatz an dem Bau der chinesischen Mauer beim Kaiser Ruhm erlangen. Jeder Spieler möchte der erfolgreichste Bauherr sein.

Es liegen mehrere Paare von Ruhmeskärtchen aus. Die Spieler wetteifern um die Ruhmeskärtchen, indem sie Mauerkarten auslegen. Sobald der Kartenvorrat eines Spielers aufgebraucht ist oder keine Ruhmeskärtchen mehr vorhanden sind, endet das Spiel. Sieger ist der Spieler mit den meisten Ruhmespunkten, die jeweils auf den Ruhmeskärtchen stehen.

1.2.3 Spielvorbereitung

Je nach Spieleranzahl werden zufällig Paare aus Ruhmeskärtchen, welche eine Wertigkeit zwischen 1 und 8 besitzen, ausgelegt. Jedes Paar markiert einen „Bauabschnitt“.

2 Spieler: 2 Bauabschnitte

3 Spieler: 3 Bauabschnitte

4 – 5 Spieler: 4 Bauabschnitte

Abbildung 3: Spielaufbau mit 2 Bauabschnitten

Jeder Spieler mischt seinen eigenen Kartenstapel aus je 20 Karten, zieht zu Beginn 5 Karten davon und legt die anderen verdeckt als Nachziehstapel ab.

1.2.4 Spielverlauf

Die Spieler sind nacheinander am Zug. Dabei führt man drei Aktionen durch:

1. Aktion: Prüfen, ob man an einem Bauabschnitt (oder an mehreren) mehr Punkte besitzt als alle anderen Spieler (siehe Wertung)
2. Aktion: Entweder eine Karte vom Stapel nachziehen oder eine einzelne oder mehrere identische Karten aus der Hand offen an einen beliebigen Bauabschnitt anlegen.
3. Aktion Wie 2. Aktion

Beispiel:

In einem Spiel mit zwei Spielern hat der erste Spieler die Farbe Rot gewählt und hat nach den Spielvorbereitungen seine fünf Karten auf der Hand. Er betrachtet nun das Spielfeld: (Hinweis: Die Karten des Spielers liegen normalerweise nicht offen. Das dient nur der Erklärung in diesem Kapitel)

Abbildung 4: Der rote Spieler ist am Zug

1. Aktion:

Als erste Handlung in seinem Zug prüft der Spieler die Kräfteverhältnisse an allen Baureihen. Da noch keine Karten ausliegen, entfällt die Prüfung.

2. Aktion:

Der Spieler kann nun eine Karte ziehen, oder Karten aus seiner Hand anlegen. Der Spieler möchte die beiden normalen Mauerkarten mit dem Wert eins (ohne aufgedruckten Kämpfer) an die erste Reihe legen. Da sie identisch sind, kann er sie zusammen in einer Aktion legen.

3. Aktion:

Der Spieler legt die Turmkarte mit dem Wert drei an die zweite Baureihe an. Er kann keine anderen Karten dazu auslegen, da die Karten dazu exakt identisch sein müssen.

Abbildung 5: Spielfeld nach Aktion 2

Abbildung 6: Spielfeld nach Aktion 3

1.2.5 Wertung

Falls ein Spieler zu Beginn seines Zuges an einem Bauabschnitt mehr Punkte besitzt als alle anderen Spieler (kein Gleichstand), oder als einziger Spieler dort Karten liegen hat, so erhält er zum Lohn ein Ruhmeskärtchen.

Sind noch zwei Ruhmeskärtchen vorhanden, wählt der Spieler zwischen ihnen und legt eines davon offen auf eines seiner Karten im Bauabschnitt.

Ist nur eines vorhanden, so nimmt es der Spieler an sich und legt es verdeckt bei sich ab. Die Baureihe wird nun aufgelöst und: Das Ruhmeskärtchen, das bereit auf einer Karte im Bauabschnitt liegt, geht an den Besitzer dieser Karte. Danach werden alle Karten des Bauabschnitts entfernt und zwei neue Ruhmeskärtchen ausgelegt, falls noch genügend vorhanden sind.

Um die Punkte eines Spielers in einem Bauabschnitt zu bestimmen, werden normalerweise die Punktwerte seiner Karten zusammengezählt. Sonderregeln für die einzelnen Karten finden Sie im Abschnitt „Die einzelnen Karten und ihre Bedeutung“. Falls auf einer der Karten des Spielers ein Ruhmeskärtchen liegt, so wird der Wert auf diesem Ruhmeskärtchen von der Punktzahl des Spielers **abgezogen**.

Beispiel:

Nach ein paar Zügen ist wieder der rote Spieler am Zug. Das Spielfeld sieht am Anfang seines Zuges so aus:


Abbildung 7: Spielfeld am Zugbeginn

Der rote Spieler führt nun seine erste Aktion durch und prüft die Kräfteverhältnisse an allen Baureihen. An der oberen Reihe hat er zwei Karten, die jeweils einen Punkt bringen, der blaue Spieler ist dort jedoch mit einer Karte vertreten, die zwei Punkte wert ist. Somit herrscht ein Gleichstand von zwei zu zwei Punkten. Das bedeutet, dass der rote Spieler dort kein Ruhmeskärtchen gewinnt.

An der unteren Reihe hat der rote Spieler eine Karte mit dem Wert drei ausliegen, der blaue Spieler eine Karte mit dem Wert eins. Der rote Spieler ist somit der stärkste Spieler an dieser Reihe und darf sich ein Ruhmeskärtchen aussuchen, welches er auf seine Karte legt. Er wählt das höhere Ruhmeskärtchen.


Abbildung 8: Spielfeld nach dem Zug

Der rote Spieler zieht mit seiner 2. und 3. Aktion zwei Karten nach. Nun ist der blaue Spieler an der Reihe. An der ersten Reihe herrscht noch immer Gleichstand. An der zweiten Reihe besitzt der blaue Spieler einen Punkt, der rote Spieler hat dort zwar eine Karte mit dem Wert drei liegen, davon werden jedoch acht Punkte von dem darauf liegenden Ruhmeskärtchen abgezogen. Somit hat er -5 Punkte.

Der blaue Spieler ist also der stärkste Spieler an der Reihe. Er nimmt sich das verbleibende Märkchen mit dem Wert vier und legt es verdeckt vor sich ab. Der rote Spieler legt das Märkchen mit dem Wert 8, das auf seiner Karte liegt, und legt es ebenso vor sich ab. Nun werden alle Mauerkarten der unteren Reihe aus dem Spiel genommen und zwei neue Ruhmeskärtchen ausgelegt.

1.2.6 Die einzelnen Karten und ihre Bedeutungen

Es gibt sieben verschiedene Kartensorten. Die ersten drei besitzen keine besondere Wirkung:


7 Mauern

Zählt 1 Punkt.


3 Tore

Zählt 2 Punkte.


1 Wachturm

Zählt 3 Punkte.

Die anderen vier Karten haben zwar auch einen Punktwert, der in die Wertung einfließt, aber vor allen Dingen zeichnen sie sich durch eine spezielle Wirkung aus:

1 Adeliger

Zählt 1 Punkt.

Der Adelige verändert den Wert aller Mauerkarten in seiner Baureihe auf eins. Er wirkt sowohl auf eigene Karten wie auch auf die Karten anderer Spieler. Ein einzelner Adeliger hat dieselbe Wirkung wie mehrere Adelige.

5 Kämpfer

Kämpfer werden stärker, je mehr von ihnen zusammen an einem Abschnitt liegen. Der erste Kämpfer zählt einen Punkt. Der zweite Kämpfer (von der selben Farbe) zählt bereits zwei Punkte. Der dritte zählt drei Punkte und so weiter. Falls ein Adeliger in der Reihe liegt, ist jedoch jeder Kämpfer nur einen Punkt wert, egal, wie viele Kämpfer es sind.

Abbildung 9: Wirkung des Adeligen

Abbildung 10: Wirkung des Kämpfers

2 Reiter

Zählt 2 Punkte.

Der Reiter kann ausgelegt werden, ohne dabei eine Aktion zu verbrauchen.

1 Drache

Zählt 1 Punkt.

Der Drache kann wie jede andere Karte an einen Bauabschnitt angelegt werden, jedoch kann er auch auf eine beliebige Karte des Gegners gelegt werden, außer, wenn ein Ruhmeskärtchen darauf liegt. Falls der Drache eine andere Karte überdeckt, zählt sie nicht mehr und erzielt auch keine Wirkung mehr.

Abbildung 11: Wirkung des Drachen

1.2.7 Spielende

Wird bei einer Wertung das letzte Ruhmeskärtchen des Spiels gewonnen, so endet das Spiel sofort.

Falls ein Spieler seine letzte Karte ausgespielt hat, und auch keine Karten mehr zum Nachziehen hat, endet sein Zug, und es beginnt die letzte Runde. Alle anderen Spieler sind noch einmal an der Reihe. Ist der Spieler wieder am Zug, der keine Karten mehr hat, darf kein Spieler mehr Karten anlegen. Das Spiel geht jedoch reihum weiter, so dass Spieler noch Wertungen durchführen und Ruhmeskärtchen gewinnen können, bis keine Wertung mehr durchzuführen ist, weil alle Ruhmeskärtchen gewonnen sind, oder an manchen Bauabschnitten die Spieler gleich viele Punkte haben.

Sobald das Spiel beendet ist, zählt jeder Spieler die Punkte seiner gewonnenen Ruhmeskärtchen zusammen. Ruhmeskärtchen, die noch auf Karten im Spiel liegen, zählen hier nicht dazu!

Der Spieler, der die höchste Punktzahl erreicht hat, ist der Sieger.

1.3 Warum gerade dieses Spiel?

Das Spiel besteht sowohl aus Taktik als auch aus Glückselementen. Die Karten, die einem zur Auswahl stehen, werden von einem gemischten Stapel gezogen und sind daher zufällig sortiert. Allerdings stehen dem Spieler zahlreiche verschiedene taktische Möglichkeiten zur Verfügung, die das Spiel für mich interessant machen. Man muss abwägen, wann man Karten nachzieht, und wann man statt dessen Karten auslegt. Mehr Karten in der Hand ergeben eine größere Auswahl und somit mehr Möglichkeiten, außerdem kann man so öfter mehrere gleiche Karten zugleich spielen. Jedoch ist man in der Zeit, in der man Karten nachzieht, passiv, und lässt den Gegnern Raum zum Punkten. Aber auch wenn man aktiv wird muss man abschätzen, wo man welche Karten anlegen sollte. Und wenn man an einer Reihe gewonnen hat, bleibt auch noch die Entscheidung: Nimmt man das hohe Märkchen, um sich sofort die hohe Punktzahl zu sichern, oder doch das niedrigere, um auch eine Chance auf das zweite Märkchen zu haben?

All dies motivierte mich zur Entwicklung eines künstlichen Gegenspielers. Das Spiel ist nicht primär vom Glück abhängig, wie beispielsweise „Mensch-ärgere-dich-nicht“. Man kann die Möglichkeiten seiner Gegenspieler aber auch nicht komplett berechnen, wie beispielsweise beim Schach. Ein Algorithmus, der auf jeden möglichen eigenen Zug die Möglichkeiten der Gegner durchrechnet, wie es beim Schach üblich ist, gestaltet sich hier schwierig, da die Kartenauswahl der Gegner zu ungewiss ist. Man müsste den durchschnittlich besten Zug für alle möglichen Kartenverteilungen bei den Gegnern ermitteln, und diese Möglichkeiten sind einfach zu zahlreich. Es gibt für diese Vorgehensweise Wege, um dieses Problem zu lösen. So wird eine große Anzahl von Kartenverteilungen zufällig generiert, für diese Situationen jeweils der beste Zug ermittelt, und nun der Zug benutzt, der im Schnitt am Besten funktioniert hat. Ich habe mir jedoch als Ansatz den evolutionären Algorithmus gewählt, den ich im folgenden Kapitel vorstellen möchte.

2 Evolutionärer Algorithmus

In diesem Kapitel wird kurz die Funktionsweise des Algorithmus im Allgemeinen beschrieben und verschiedene Möglichkeiten erklärt. Es wird noch kein Bezug auf das aktuelle Problem genommen, dies folgt im nächsten Kapitel.

2.1 Grundlagen

„Ein Evolutionärer Algorithmus ist ein Optimierungsverfahren, das als Vorbild die biologische Evolution hat.“ (Quelle: Wikipedia)

Einem Evolutionären Algorithmus liegt immer ein „Erbgut“ zu Grunde. Dieses Erbgut ist problemspezifisch und stellt ganz allgemein eine gültige Lösung eines Optimierungsproblems dar, also eines Problems, zu welchem man in absehbarer Zeit eine gute Lösung finden möchte – wenn möglich, die absolut beste Lösung, das „globale Optimum“. Allgemein gesagt wird versucht, eine Startlösung so zu verändern, dass sie immer besser und besser wird, indem viele ähnliche Lösungen mit der ursprünglichen verglichen werden, und unter ihnen die Beste ausgewählt wird. Danach fährt man die Suche bei der neuen Lösung fort, so lange, bis ein Abbruchkriterium erfüllt ist. Im guten Fall schließt die Suche bei einem „lokalen Optimum“ ab, also einer Lösung, die man nicht mehr durch ein paar kleine Änderungen verbessern kann. Ideal wäre, wenn diese Lösung auch das „globale Optimum“ ist.

Ein Evolutionärer Algorithmus besteht aus mehreren Teilschritten, die jeweils dem Problem angepasst werden. Einen groben Ablauf beschreibt folgendes Diagramm. (Grundlagen dafür aus Quellen [4] und [5])

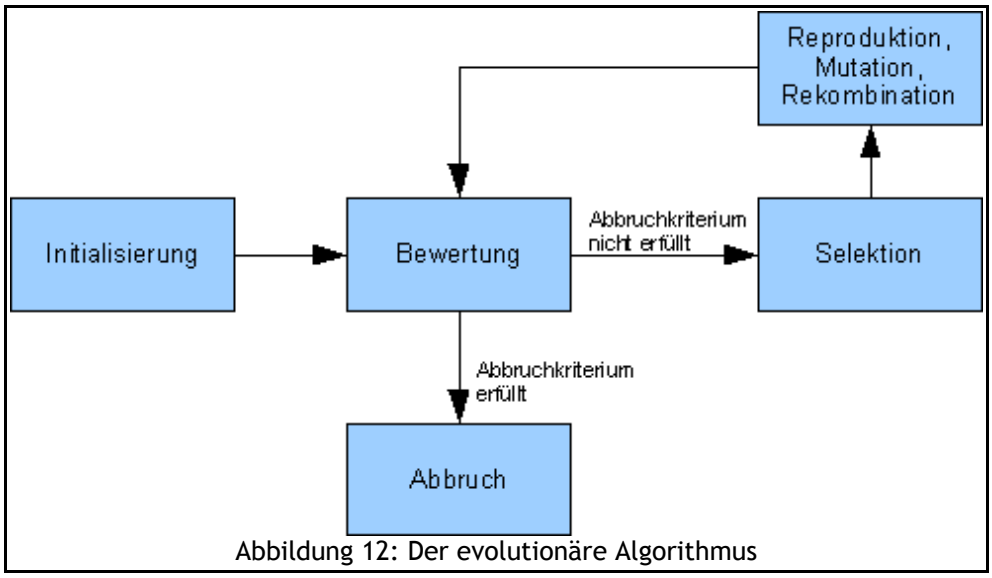

Abbildung 12: Der evolutionäre Algorithmus

Die einzelnen Teile werden im Folgenden genauer erklärt.

2.2 Initialisierung

Der Algorithmus funktioniert im Ansatz von jedem beliebigen Startwert, also jeder gültigen Lösung. Jedoch kann die Wahl des Startpunktes unter Umständen große Auswirkungen auf die Qualität der resultierenden Lösung haben. Von manchen Startpunkten ist es vielleicht sehr wahrscheinlich, dass man bei einem bestimmten lokalen Optimum landet, welches jedoch, verglichen mit dem globalen Optimum, nur sehr schlecht ist. Ansätze wären die geschickte Wahl eines Startwerts, falls dies möglich ist, oder die Wiederholung des Algorithmus mit vielen verschiedenen Startwerten, die auch zufällig gewählt werden können. Die resultierenden Lösungen kann man anschließend vergleichen und die Beste daraus bestimmen.

2.3 Bewertung

Es ist wichtig und nicht immer einfach zu definieren, was eine gute und was eine schlechte Lösung ist. Es müssen Kriterien aufgestellt werden, die jede Lösung bewerten und gegeneinander abwägen.

Einen Wurf beim Kegeln zu bewerten ist beispielsweise noch recht einfach, da ein Wurf besser ist, je mehr Pins umgeworfen wurden. Es bleibt noch zu klären, wie man zwei Würfe mit gleicher Pin-Anzahl bewertet. Falls die übrigen Pins sehr weit auseinander stehen ist dies ungünstig. Besser wäre es, wenn die Pins zusammen stehen.

Sehr viel komplexer ist aber die Bewertung eines Zuges beim Schach. Ein Zug, bei dem eine gegnerische Figur geschlagen wird, ist nicht zwangsläufig gut. Man muss stets abwägen, welche Möglichkeiten der Gegner hat, auf den eigenen Zug zu antworten, und es muss die Stellung aller Figuren zueinander betrachtet werden. All diese Gedanken müssen in eine Bewertungs-Funktion einfließen, aus der ein möglichst eindeutiger Wert heraus kommt, den man nun vergleichen kann.

Dieses Ergebnis kann für das Abbruchkriterium benutzt werden, mit dem entschieden wird, wann der Algorithmus stoppen soll. Man kann also beispielsweise eine Grenze angeben, die bestimmt, wann eine Lösung für „gut genug“ befunden wird. Sobald diese Grenze erreicht ist, wird die gefundene Lösung übernommen.

2.4 Mutation

Die Mutation hat die Aufgabe, aus bestehenden Lösungen neue Lösungen zu generieren, indem zufällige Veränderungen vorgenommen werden. Die Mutationen sollen „ungerichtet“ sein, also keinem bestimmten Ziel folgen. Es werden die Auswirkungen der Mutation bewertet, nicht die Mutation selbst.

Die Mutation-Strategie sollte so gewählt werden, dass von einer Ausgangslösung durch mehrmalige Mutation jede beliebige andere (gültige) Lösung erreicht werden kann.

Soll man zum Beispiel zu einer gegebenen Menge von Städten die kürzeste Flugroute über alle Städte errechnen, besteht eine Lösung aus einer beliebigen Route, die alle Städte genau ein Mal beinhaltet. Eine mögliche Mutations-Strategie wäre, einfach zwei zufällige Städte in der Reihenfolge zu tauschen. So entsteht eine neue Reiseroute, die nun bewertet werden muss. Durch mehrmaliges Tauschen kann jede beliebige Reihenfolge erzielt werden.

2.5 Rekombination

Eine weitere Methode, um aus alten Lösungen neue Lösungen zu erzeugen, ist die geschickte Verknüpfung zweier oder mehrerer Ausgangslösungen. Zum Beispiel werden zwei Lösungen miteinander so verknüpft, dass die resultierende Lösung zu gleichen Teilen aus beiden Lösungen besteht, und wieder eine gültige Lösung ergibt. Man könnte den ersten Teil der Lösung von einer Ausgangslösung und den zweiten Teil aus einer anderen Ausgangslösung übernehmen. Oder man würfelt bei jedem Teil der Lösung, aus welchem „Elternteil“ der Teil übernommen werden soll. Dies entspricht in etwa dem Vorgang der genetischen Rekombination von zwei Erbmaterialien.

Rekombination ist nicht immer notwendig. In manchen Fällen reicht es, wenn die Mutationen genügend unterschiedliche Lösungen erzeugen.

2.6 Selektion

Nachdem durch Mutation und Rekombination verschiedene Lösungen erzeugt wurden, müssen aus dieser Auswahl die besten Lösungen bestimmt werden. Die geschieht mit Hilfe der Bewertungs-Funktion. Je nach Strategie wird so immer die beste Lösung gewählt, oder es werden schlechtere Lösungen zugelassen, um sich so aus lokalen „Sackgassen“ zu befreien. Es ist möglich, dass nur eine Lösung ausgewählt wird, oder aber eine bestimmte Menge von Lösungen.

3 Entwicklung des künstlichen Spielers

Dieses Kapitel befasst sich mit der Programmierung und den verschiedenen Ansätzen, die bei der Entstehung getestet wurden. Es werden Vor-/Nachteile und entstandene Schwierigkeiten erwähnt.

3.1 Programmierung

Das gesamte Programm wurde mit der Programmiersprache Java (Version 1.6.0 Update 2) mit Hilfe der Eclipse Entwicklungsumgebung geschrieben. SWT Klassen wurden zur optischen Darstellung und Bedienung des Spiels verwendet.

Das Programm läuft entweder im Benutzer-Modus, in dem ein oder mehrere Spieler gegen computergesteuerte Gegner antreten können. Der Algorithmus ist während diesen Spielen inaktiv. Alternativ ist das Programm im Entwicklungs-Modus. In diesem Modus sind sämtliche grafischen Elemente entfernt, so dass man das Spielfeld nicht sehen kann. Die Elemente werden nicht unsichtbar geschaltet, sondern erst gar nicht erstellt, da sonst der Speicher nur unnötig belastet werden würde, und so die Rechenzeit viel zu hoch wäre. Der Modus dient zum Training und zur Überprüfung des künstlichen Spielers. Es können beliebig viele Testspiele pro Generation und dazu beliebig viele Generationen durchlaufen werden, dazwischen wird in definierbaren Abständen die Spielstärke des aktuellen Spielers mit Hilfe von Kontrollrunden überprüft. Der Entwicklungsmodus soll von einem normalen Benutzer nicht verfügbar sein, sondern nur vom Entwickler gestartet werden können.

3.2 Funktionsweise des Gegenspielers

Sobald ein computergesteuerter Spieler am Zug ist, hat er folgende Situation zu bewältigen:

- Er hat eine bestimmte Anzahl und Auswahl an Karten auf der Hand
- Die erste Aktion, die Überprüfung aller Bauabschnitte, braucht nahezu kein Eingreifen seitens des Spielers. Lediglich falls der Spieler das erste Märkchen an einer Reihe gewinnt, bleibt zu entscheiden, welches Märkchen er nimmt und auf welche eigene Karte er es legt. Letztere Entscheidung ist trivial; die Karte, auf die man das Märkchen legt, ist vor feindlichen Angriffen von Drachen geschützt. Deshalb fällt die Wahl immer auf das Märkchen mit dem aktuell höchsten Wert. Die Entscheidung, welche Marke genommen wird, fällt auf Grund des aktuellen Vorsprungs vor den anderen Spielern und dem Wert der Marken. Falls man nach Aufnahme des niedrigeren Märkchens noch immer der Führende ist, oder zumindest einen Gleichstand an der Spitze erzielt, so nimmt man dieses Märkchen. Ansonsten das höhere. Dieses Verhalten erscheint mir relativ simpel und logisch, so dass es außerhalb des Algorithmus fest geschrieben und für alle Gegenspieler gleich ist.
- Danach besitzt der Spieler zwei Aktionen. Es muss zwischen dem Nachziehen vom Stapel und dem Auslegen eines oder mehrerer Karten entschieden werden, so lange, bis keine Aktionen mehr übrig sind.

```
public void MakeYourMove() {

    // Erste Aktion: Kräfteverhältnisse prüfen
    BS.checkMyInfluenceAI();

    // Nun müssen zwei Aktionen durchgeführt werden
    RemainingActions = 2;

    while (RemainingActions > 0) {
        doAction();
    }

    MoveFinished();
}
```

Abbildung 13: Codebeispiel - Zug des Gegenspielers

Dieser Entscheidungsprozess wird von einer Reihe Algorithmen gesteuert. Um den Entstehungsprozess dieser Algorithmen aufzuzeigen, stelle ich zuerst die ursprüngliche Version des Programms vor. Danach erläutere ich, warum diese Version überarbeitet werden musste, und schließlich werden die Änderungen zur neuen Version vorgestellt.

3.2.1 Alte Version

In der früheren Version wurden die Entscheidungen über die Vorgehensweise in den Aktionen einzeln für jede Aktion bestimmt. Dazu wurden alle möglichen Züge durchprobiert und das Ergebnis jeweils bewertet. Alle möglichen Züge bedeutet:

- Jede Sorte von Karten, die in der Hand vorhanden sind
- In jeder möglichen Anzahl
- An jede mögliche Reihe anlegen.

Die Möglichkeiten, die Drachenkarte auf verschiedene Art und Weisen zu benutzen, wird hierbei außer Acht gelassen. Es wird wieder ein fixer Algorithmus benutzt, da die optimale Nutzung der Karte meiner Ansicht nach einfach ist. Man legt die Karte

immer auf die höchste Karte des führenden Spielers, so erzielt man den meisten Nutzen.

Zugbewertung:

Um die Züge nun zu bewerten, wurde die Situation an der betreffenden Reihe vor dem Zug mit der Situation an der Reihe nach dem Zug bewertet, indem der Zug einfach simuliert wurde. Die Bewertung betrachtete nun beispielsweise:

- Hat sich die Position des Spielers durch den Zug verbessert?
- Auf die erste Position?
- Mit wie großer Führung?
- Wie lukrativ sind die zu gewinnenden Märkchen an der Reihe?

Für jedes Bewertungs-Kriterium enthielt der Zug eine gewisse Anzahl von Punkten, die addiert wurden. Aus allen möglichen Zügen wurde der beste ausgewählt. Überschritt dessen Punktzahl einen gewissen Schwellwert, so wurde der Zug ausgeführt. Falls der Schwellwert nicht erreicht wurde, wurden alle möglichen Züge als zu schlecht befunden. In diesem Fall wurde eine Aktion dazu benutzt, eine Karte nachzuziehen. Dieser Vorgang wurde so lange wiederholt, bis alle Aktionen aufgebraucht waren. Folgendes Diagramm soll diese Vorgehensweise noch einmal verdeutlichen:

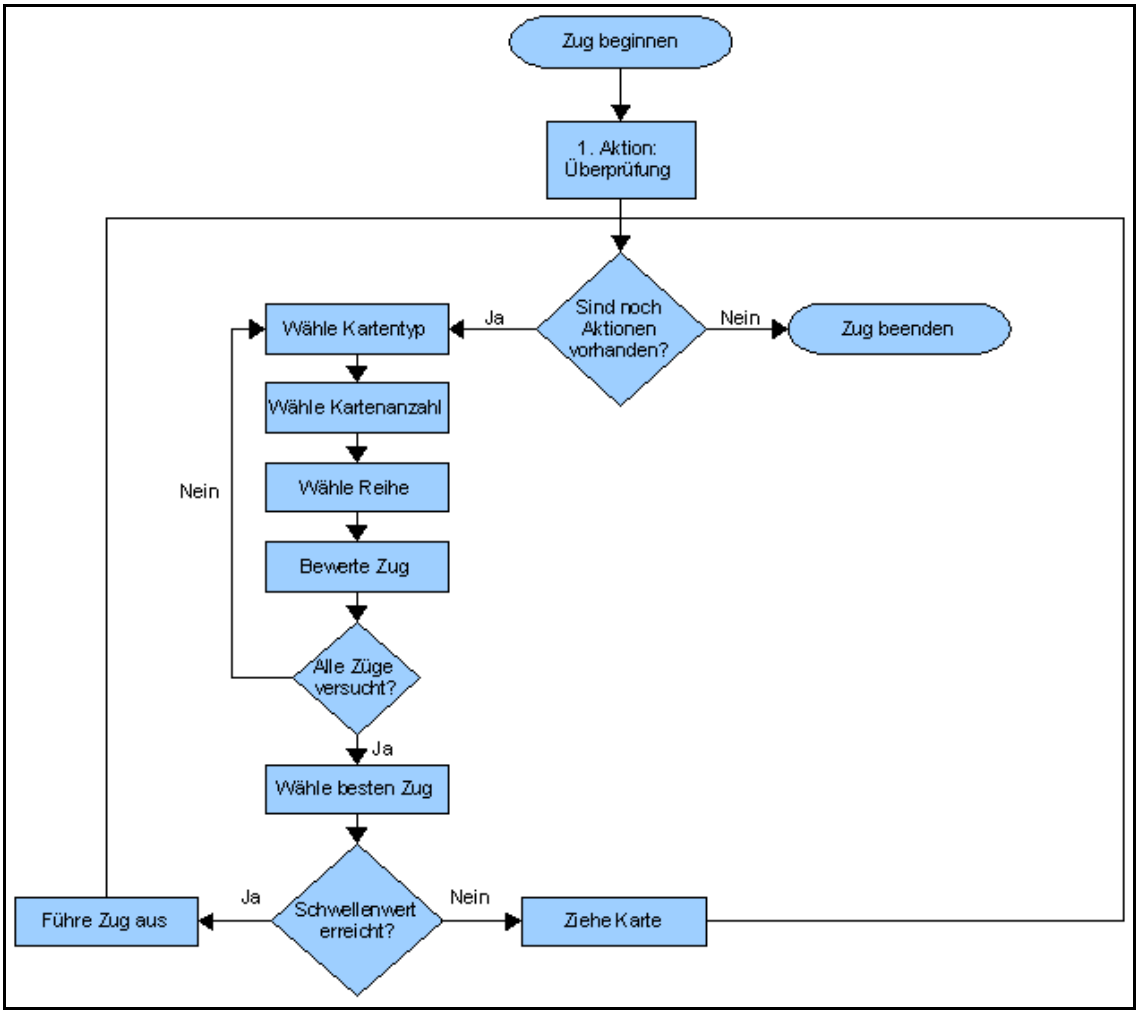

Diagramm 1: Ablauf Alte Version

Die Anzahl der Zugbewertungen bestimmt letztendlich die Zeit, die der Algorithmus braucht. Alle anderen Schritte werden weit weniger oft durchlaufen und nehmen daher wenig Rechenzeit in Anspruch.

Um eine Vorstellung darüber zu haben, wie viele Züge in etwa pro Durchlauf der Zuggenerierung und letztendlich pro Spiel generiert und durchprobiert wurden, folgen nun einige Rechnungen.

Anzahl der Zugmöglichkeiten:

Der schlechteste Fall tritt ein, falls der Spieler alle 20 Karten bei einem Spiel mit vier Baureihen auf seiner Hand hält. In diesem Fall gäbe es

20 * 4 = 80 Bewertungen pro Durchlauf.

Die 20 lässt sich so erklären: Mit 20 gleichen Karten hätte man exakt genau so viele verschiedene Möglichkeiten, Karten an einer Reihe anzulegen, wie mit 20 unterschiedlichen Karten. Bei 20 unterschiedlichen hat jeweils die Möglichkeit, jede Karte einzeln zu legen, das ergibt 20 Möglichkeiten. Bei 20 gleichen Karten hat man die Möglichkeit, eine Karte zu legen, oder zwei, oder drei... und so weiter. Das ergibt auch 20 Möglichkeiten, da es bei gleichen Karten keine Rolle spielt, welche Karten man benutzt. Genauso verhält es sich bei jeglichen Mischungen. Ob ich auf der Hand drei Karten einer Sorte und vier Karten einer anderen halte, ob alle sieben Karten gleich sind oder alle verschieden sind, es gibt immer sieben Möglichkeiten.

Nun kommt es aber praktisch nie vor, dass ein Spieler alle 20 Karten seines Stapels auf der Hand hält. Ich halte nach intensivem Spiel fünf Karten für einen realistischen Durchschnittswert. Das ergäbe bei einem Spiel mit vier Baureihen:

5 * 4 = 20 Bewertungen pro Durchlauf.

Bei einem Spiel mit zwei Baureihen sind es dem entsprechend 10 Bewertungen.

Durchläufe pro Spieler:

Wir nehmen an, jeder Spieler könnte seine Hand komplett leer legen. Die maximale Anzahl an Durchläufen der Zug-Generierung beträgt in diesem Fall:

15 Aktionen

+ 20 Aktionen, bei denen jede Karte einzeln abgelegt wird

= 35 Durchläufe pro Spieler.

Ein Minimal-Wert ergibt sich, wenn man jede Kartensorte geschlossen ablegt:

15 Aktionen, die zum Nachziehen der Karten verwendet werden

+ 7 verschiedene Kartensorten, die vollständig angelegt werden

= 22 Durchläufe pro Spieler.

Realistisch sind beide Fälle nicht, daher nehme ich einen Mittelwert von 30 Durchläufen an.

Zug-Bewertungen pro Spiel:

Die Menge der Zugbewertungen hängt stark von der Anzahl der Spieler ab. Unter der Annahme, dass beide Spieler ihre Karten vollständig ausspielen, und unter der Verwendung der angenommenen Mittelwerte ergibt sich für zwei Spieler:

10 Bewertungen pro Durchlauf

* 30 Durchläufe pro Spieler

* 2 Spieler

= 600 Bewertungen pro Spiel.

Für vier Spieler ergibt die Rechnung:

20 Bewertungen pro Durchlauf

* 30 Durchläufe pro Spieler

* 4 Spieler

= 2400 Bewertungen pro Spiel.

Testläufe zeigen allerdings, dass diese Werte zu hoch sind. Für ein Spiel mit vier Spielern wurden im Schnitt 1500 Bewertungen festgestellt.

Dieser Algorithmus ist jedoch zu simpel. Die große Schwachstelle ist, dass er keinen einzigen Zug im Voraus planen kann, und niemals einen vorbereitenden Zug durchführt, der selbst noch keine große Wirkung hat. Deshalb wurde der Algorithmus verändert.

3.2.2 Neue Version

Die alte Version wurde dahingehend verändert, dass die Zuggenerierung nun nicht mehr nur einzelne Züge erstellt, sondern Zugfolgen, und sie sich selbst rekursiv aufrufen kann. Diagramm 5 auf Seite 22 zeigt den Ablauf der Zuggenerierung. Sie arbeitet nun folgendermaßen:

Es werden, wie auch vorher, alle möglichen Züge generiert, das bedeutet:

- Jede Sorte von Karten, die in der Hand vorhanden sind
- In jeder möglichen Anzahl
- An jede mögliche Reihe anlegen.

Diese Züge werden erst einmal bewertet. Doch danach wird geprüft, ob nach dem Zug noch weitere Aktionen übrig wären. Ist dies der Fall, ruft sich die Zuggenerierung selbst nochmal auf und übergibt eine Liste mit dem bisherigen Zugverlauf. Im Diagramm ist dies der Prozess unten rechts: Zuggenerierung (Aktuelle Zugfolge). Es werden nun also alle Züge, die nach dem ersten Zug möglich sind, durchlaufen, und

dem ersten Zug angehängt. Nun wird die Zugfolge als Ganzes bewertet. Und auch danach wird geprüft, ob noch weitere Züge angehängt werden können.

Mehr als zwei Züge hintereinander sind nur durch den Einsatz von Reiter-Karten möglich, da diese keine Aktion verbrauchen. Da es zwei Reiter für jeden Spieler gibt, kann es Zugfolgen mit einer maximalen Länge von vier Zügen geben.

Durch Zugfolgen ergeben sich neue Möglichkeiten, denn es könnte sein, dass ein Zug alleine eine sehr schlechte Bewertung erhält, mit einem günstigen zweiten Zug jedoch großen Nutzen bringt.

Zugbewertung:

Die Zugbewertung musste auf Grund dieser Änderungen angepasst werden. Es wird nicht mehr die Situation an einer bestimmten Reihe betrachtet, sondern die Gesamtsituation des Spielers an allen Reihen. Diese wird vor der Generierung der Zugmöglichkeiten einmalig ermittelt, bewertet und gespeichert. Die Bewertung ist ähnlich wie in der alten Version und betrachtet folgende Faktoren:

- Besteht eine Führung an einer Reihe?
  - Wie lukrativ sind die Märkchen an dieser Reihe?
  - Wie groß ist die Führung?
- Besteht ein zweiter Platz an einer Reihe?
  - Wie lukrativ ist das kleinere Märkchen an dieser Reihe?
  - Wie groß ist der Abstand zu den anderen Spielern?

```
// Erster Platz Alleine?
// Bei hohem erstem Token??
// Mit großem Vorsprung???

if ((myScore > 0) && (myScore == Highscore) && (AnzahlErster == 1)) {
    Bewertung += GewichtErsterPlatz;
    Bewertung += (GewichtHohesToken * erstesToken);
    Bewertung += (GewichtErsterVorsprung * (Highscore - SecondHighscore));
}
```

Abbildung 14: Codebeispiel - Ausschnitt aus Bewertungs-Funktion

Bei der Zuggenerierung wird die Gesamtsituation nach der Zugfolge simuliert, bewertet und mit der Ausgangs-Bewertung verrechnet.

Wichtig sind dabei folgende Faktoren:

- Wie sehr hat sich die Punktzahl verbessert?
- Wie viele Karten waren nötig, um die Zugfolge zu spielen?
- Wie viele Aktionen waren nötig, um diese Zugfolge zu spielen?

Diese Überlegungen sind notwendig, um sinnlose Züge zu vermeiden. Man könnte ja beispielsweise, statt zwei gleichen Karten an eine Reihe zu legen, diese Karten einzeln an der Reihe ausspielen. Das Ergebnis wäre das Gleiche und die Bewertung deshalb die Selbe. Doch die zweite Variante verbraucht statt einer Aktion zwei Aktio-

nen, deshalb muss sie mit einer Strafe belegt werden, damit die Gesamtbewertung schlechter ausfällt.

Nachdem alle möglichen Züge und Zugfolgen bewertet wurden, wird, wie bei der vorherigen Version, der beste Zug ermittelt und mit einem Schwellenwert verglichen. (Siehe Diagramm 4 – untere Verzweigung: „Schwellenwert erreicht?“) Daraus wird entschieden, ob die Zugfolge ausgeführt oder eine Karte nachgezogen wird. Sind danach noch Aktionen übrig, so wird ein weiterer Durchlauf gestartet.

Die Diagramme zeigen in zwei Teilen den beschriebenen Algorithmus. Das erste Diagramm veranschaulicht die Prozedur, die sich selbst rekursiv aufrufen kann. Das zweite Diagramm zeigt den gesamten Ablauf eines Zuges.

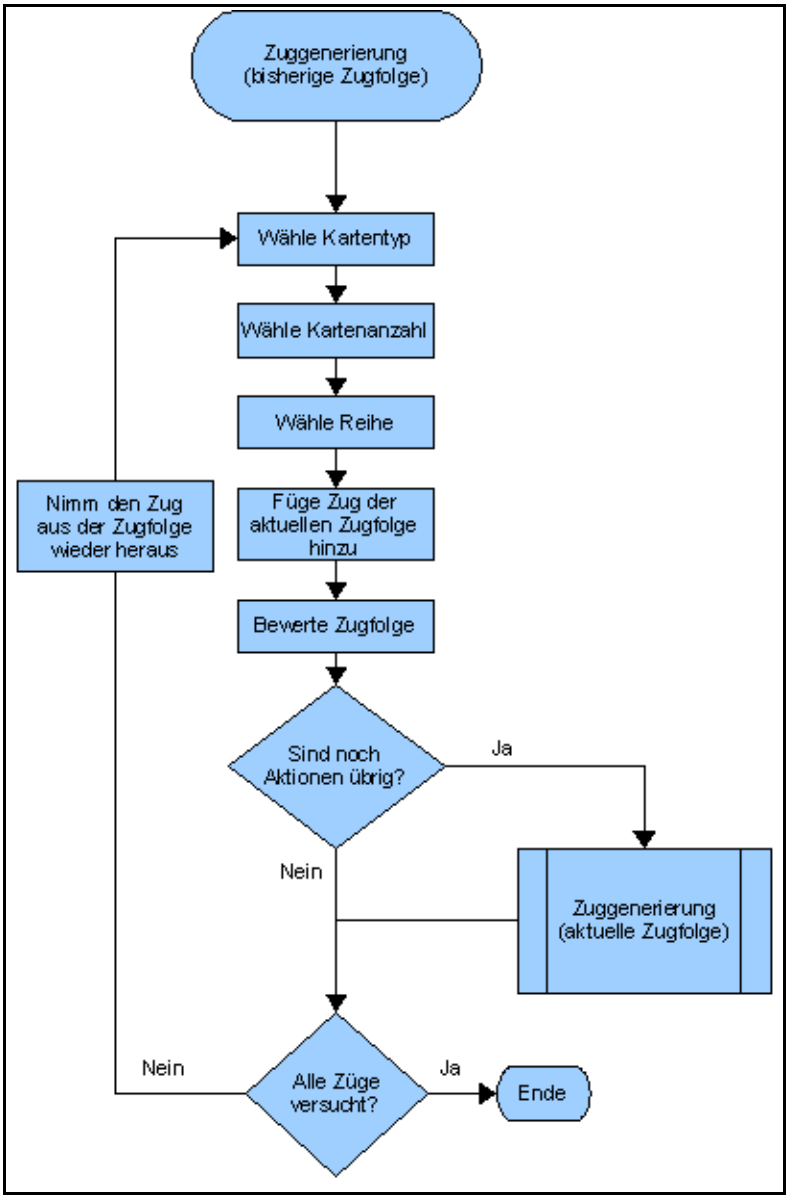

Diagramm 2: Rekursive Prozedur: Zuggenerierung

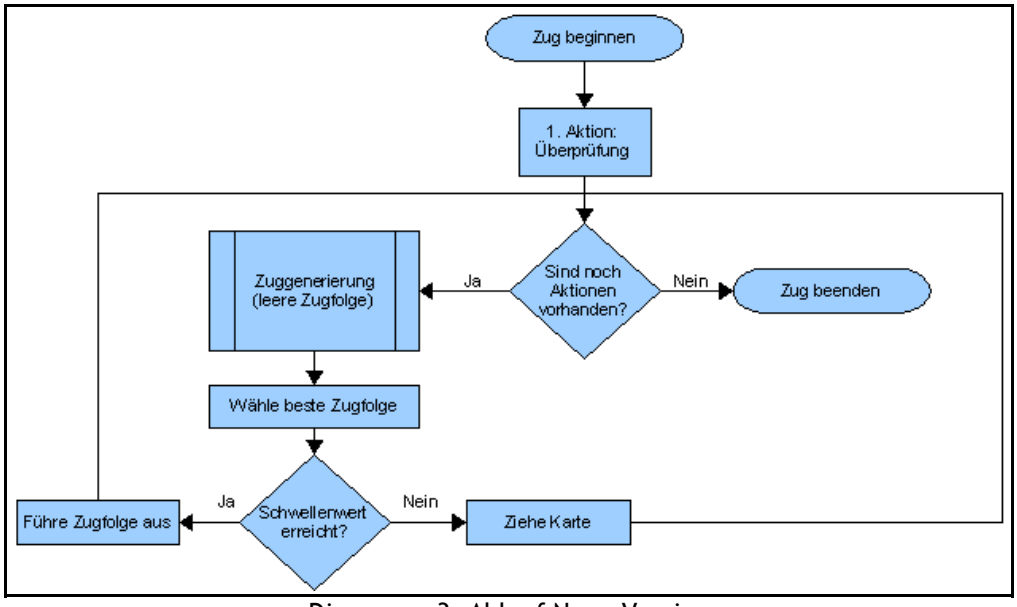

Diagramm 3: Ablauf Neue Version

Anzahl der Zugmöglichkeiten:

Durch die Betrachtung von Zugfolgen gewinnt der künstliche Spieler klar an Spielstärke, jedoch entsteht ein enorm hoher Rechenaufwand, da sehr viel mehr Zugmöglichkeiten bei jedem Durchlauf betrachtet werden.

Nehmen wir als Beispiel an, dass bei einem Spiel mit vier Baureihen ein Spieler am Zug ist, der noch zwei Aktionen übrig hat, und fünf unterschiedliche Karten auf der Hand hält, von denen keine ein Reiter ist.

Zählt man alle Bewertungen von Zugfolgen, die aus einem Zug bestehen, so erhält man, wie schon im vorherigen Kapitel beschrieben, 5 * 4 Bewertungen.

Jede dieser Zugfolgen kann durch einen weiteren Zug ergänzt werden. Dazu gibt es, da noch vier Karten auf der Hand sind, 4 * 4 Möglichkeiten. Das macht (5 * 4) * (4 * 4) Möglichkeiten für Zugfolgen, die aus zwei Zügen bestehen, und darum eben so viele zusätzliche Bewertungen. Das ergibt insgesamt:

5 * 4 + 5 * 4 * 4 * 4 = 20 + 320 = 340 Bewertungen.

Falls auf der Hand mehrere gleiche Karten sind, wird der Wert etwas kleiner, da beim ersten Zug auch mal zwei Karten zusammen abgelegt werden können, und somit für den zweiten Zug weniger Auswahlmöglichkeiten bleiben.

Falls sich jedoch auf der Hand eine Reiter-Karte befindet, so steigt die Zahl enorm an. Bei vier Baureihen, zwei Aktionen und fünf Karten bei einer Reiter-Karte sind schon insgesamt über 1700 Bewertungen notwendig.

Bei der vorherigen Version ohne Zugfolgen gab es für dieses Beispiel pro Durchlauf nur 20 Bewertungen. Man sieht schon, dass die Laufzeit der neuen Version deutlich höher liegt.

Rechnungen über die Gesamtanzahl der Bewertungen für ein Spiel werden leider viel zu ungenau, da man Vorhersagen über die Häufigkeit von Reiter-Karten treffen muss und die Wahrscheinlichkeit für gleiche Karten in der Hand einbeziehen muss. Darum lasse ich weitere Rechnungen weg und gebe nur Werte aus Testläufen an.

Bei zwei Spielern wurden bei der alten Version etwa 300 bis 400 Bewertungen pro Spiel gemessen, bei der neuen Version schwanken die Werte zwischen 8000 und 25000 Bewertungen, im Mittel sind es etwa 8000.

Bei fünf Spieler wurden Werte zwischen 70000 und 250000 Bewertungen gemessen, bei der alten Version waren es noch etwa 1500.

Eine verblüffende Verbesserung ergab eine eigentlich einfache Änderung, mit der redundante Zugfolgen eliminiert wurden, also Zugfolgen, die es in veränderter Reihenfolge schon einmal gab. Bei Zugfolgen mit einer maximalen Länge von zwei bringt diese Verbesserung eine Halbierung der Bewertungen, jedoch werden besonders bei Durchläufen mit Reiter-Karten noch deutlich mehr Bewertungen eingespart.

Damit liegen die Werte bei zwei Spielern zwischen 3000 und 12000, bei fünf Spielern zwischen 20000 und 100000.

3.3 Repräsentation eines Individuums

Jedes Individuum des künstlichen Gegenspielers wird über die Parameter bestimmt, die zur Bewertung der Zugmöglichkeiten benutzt werden. Je nachdem, welche Werte die Individuen besitzen, führt der eine Gegenspieler einen Zug aus, auch wenn er nur wenige Punkte bringt, während der Zug bei einem anderen Spieler eine zu schlechte Bewertung erhält und somit nicht beachtet wird.

Die Parameter geben meist Punktzahlen an, die bei gewissen Gegebenheiten der Spielsituation vergeben und zusammen addiert werden, um die Situation zu bewerten.

Es gibt folgende Parameter:

- GewichtErsterPlatz: Falls der Spieler durch einen bestimmten Zug an der betreffenden Reihe alleine in Führung geht, also nach diesem Zug mehr Punkte an der Reihe besitzt als alle anderen Spieler, so wird der Parameter zur Punktzahl addiert. Ein solcher Zug ist allgemein als gut einzustufen, da er die Möglichkeit gibt, im nächsten Zug Ruhmesmärkchens zu gewinnen. Dieser Parameter ist daher sehr wichtig und wird wahrscheinlich recht hohe Werte besitzen.
- GewichtHohesToken: Falls durch den Zug eine Führung erreicht wird, also der Parameter „GewichtErsterPlatz“ zur Punktzahl addiert wurde, so wird der Wert des höheren Ruhmesmärkchens mit diesem Parameter multipliziert und zur Punktzahl addiert. Dadurch wird erreicht, dass ein Zug an einer sehr lukrativen Reihe besser bewertet wird als ein Zug an einer Reihe, an denen lediglich sehr niedrige Märkchen liegen. Der Parameter muss jedoch nicht unbedingt sehr hoch ausfallen. Ein Individuum, das hier einen recht niedrigen Wert besitzt, legt unter

Umständen öfter an niedrige Reihen an, die die anderen Spieler gar nicht beachten. Auch das kann ein Weg sein, um an Punkte zu kommen.

- GewichtErsterVorsprung: Falls durch den Zug eine Führung erreicht wird, wird außerdem noch der Abstand zum nächstbesten Spieler mit diesem Parameter multipliziert und zur Punktzahl addiert. Ein Zug, durch den man gleich einen großen Vorsprung vor seinen Konkurrenten besitzt, sollte eine sehr gute Bewertung erzielen, da man so höhere Chancen hat, dass man im folgenden Zug immer noch der Führende ist, und so Punkte gewinnen kann. Auch hier ist es nicht sicher, ob ein hoher Wert des Parameters immer gut sein muss. Ein Individuum, dass einen niedrigen Wert bei diesem Parameter besitzt, spielt meistens so, dass er knapp über seinen Gegnern liegt. Dadurch könnte er zwar auch leichter wieder überholt werden, jedoch geht er auch sparsamer mit seinen Karten um, was auch Vorteile bringen kann.
- GewichtZweiterPlatz: Falls an der betreffenden Reihe ein zweites Märkchen vorhanden ist und man durch die Ausführung des betrachteten Zuges genau so viele Punkte wie der führende Spieler der Reihe besitzt, so wird der Parameter zur Punktzahl addiert. Er wird auch addiert, falls man zwar hinter dem führenden Spieler liegt, jedoch alleine auf dem zweiten Platz. In beiden Fällen ist nämlich die Wahrscheinlichkeit gegeben, dass man noch das zweite Märkchen gewinnen kann. Dieser Parameter lässt auch Züge zu, die nicht unmittelbar eine Führung bringen. Dies kann von Vorteil sein, da man auch auf diesem Weg Punkte holen kann. Wichtig ist jedoch, dass dieser Parameter und die beiden Folgenden, die zusammen zur Geltung kommen, niedriger ausfallen als die vorigen Parameter, da sonst solche „zweiter-Platz-Züge“ schlechter bewertet werden als Züge, die eine Führung bewirken. Dies sollte nicht passieren.
- GewichtNiedrigesToken: Falls der Parameter „GewichtZweiterPlatz“ benutzt wurde, so wird der Wert des niedrigeren Ruhmesmärkchens mit diesem Parameter multipliziert und zur Punktzahl addiert. Da man davon ausgehen kann, dass der Führende Spieler bei einem Sieg das höhere Märkchen nehmen wird, was sehr oft die bessere Entscheidung ist, bleibt lediglich die Aussicht auf das zweite Märkchen. Falls dieses auch viele Punkte verspricht, so soll der Zug durch diesen Parameter gut bewertet werden.
- GewichtZweiterVorsprung: Falls der Parameter „GewichtZweiterPlatz“ benutzt wurde, so wird der Abstand zu dem drittplatzierten Spieler / den drittplatzierten Spielern mit diesem Parameter multipliziert und zur Punktzahl addiert. Je höher der Abstand zu den folgenden Spieler ist, desto höher stehen die Chancen, dass man tatsächlich das zweite Märkchen gewinnen kann.
- GewichtVerbrauchterKarten: Für jede Karte, die in diesem Zug gelegt wurde, wird dieser Wert abgezogen. Gleich gute Züge, die aber weniger Karten verbrauchen, sollen hiermit eine bessere Bewertung erhalten. Dies ist günstig, da man im nächsten Zug eine höhere Auswahl an Zugmöglichkeiten besitzt, wenn man mehr Karten auf der Hand hält.

- GewichtVerbrauchterAktionen: Für jede Aktion, die in diesem Zug verbraucht wurde, wird dieser Wert abgezogen. Auch das ist wichtig, damit keine Aktionen verschwendet werden und ein Zug mit mehreren Karten nicht auf zwei Aktionen verteilt wird, sondern zusammen als eine Aktion besser bewertet wird.
- DrawCardSchwelle: Das ist der Schwellwert, der entscheidet, ob eine Zugfolge ausgeführt oder eine Karte nachgezogen wird. Ob der Wert fest im Programm steht oder wie jetzt ein Parameter ist macht prinzipiell keinen großen Unterschied.

Jede Folge aus neun Zahlen stellt also ein gültiges Individuum dar. Das Problem ist nun die Optimierung der Zahlen, so dass ein möglichst spielstarker Gegner entsteht.

3.4 Initialisierung

Damit der Algorithmus ausgeführt werden kann, braucht man als Startpunkt ein erstes Ausgangsindividuum. Prinzipiell kann dieses Individuum beliebig sein und zufällig erstellt werden, doch dabei ergeben sich in diesem Fall einige Probleme.

Falls die Parameter so schlecht gewählt werden, dass der resultierende Spieler extreme Fehlentscheidungen macht – beispielsweise erst alle Karten vom Stapel zieht, und dann die Karten einzeln an die schlechtesten Reihen anlegt – so würde es sehr großer Änderungen an den Parametern bedürfen, um überhaupt erst einmal einen Lernerfolg feststellen zu können. Bis dann eine gewünschte Spielstärke erreicht ist, dauert es dann natürlich umso länger. Besser wäre es, wenn die Parameter schon zu Beginn nicht zu weit von einer guten Lösung entfernt wären.

Bei allen frühen Testreihen habe ich deshalb die Startwerte nach eigenem Ermessen festgelegt.

```
protected int GewichtErsterPlatz = 500;
protected int GewichtHohesToken = 100;
protected int GewichtErsterVorsprung = 100;
protected int GewichtZweiterPlatz = 250;
protected int GewichtNiedrigesToken = 50;
protected int GewichtZweiterVorsprung = 50;
protected int GewichtVerbrauchterKarten = 100;
protected int GewichtVerbrauchterAktionen = 200;
protected int drawCardSchwelle = 1000;
```

Abbildung 15: Codebeispiel - Parameter

Aktuell gibt es vor jeder Testreihe eine Initialisierungs-Runde, bei der Parameter zwar zufällig, aber doch innerhalb eigens definierter Grenzbereiche erstellt werden. Die resultierenden Spieler treten gegeneinander an und es werden die Parameter des besten Spielers als Startwert übernommen. So erhält man eine Streuung der Startwerte, die keine zu schlechten Werte zulässt. Verschiedene Startwerte sind vorteilhaft, da je nach Startwert andere Lösungen wahrscheinlicher erreicht werden können.

Im Folgenden wird beschrieben, wie aus einem Original-Individuum ein neues Individuum entsteht. In jedem Abschnitt wird dazu ein Diagramm gezeigt, welches den Verlauf dieses Vorgangs zeigt.

3.5 Mutation

Die Mutations-Strategie legt fest, wie das aktuelle Individuum verändert wird, um neue Individuen zu erzeugen. Jede Folge aus neun Zahlen ist ein gültiges Individuum. Also wird ein neues Individuum generiert, indem die alten Zahlen verändert werden.

Diese Veränderung muss zufällig sein. Es bleibt zu entscheiden, wie stark sich die neuen Individuen gegenüber ihrem Original verändern sollen. Beispielsweise könnte man lediglich einen zufällig ausgewählten Parameter und einen zufällig erzeugten Betrag verändern. Dieser Betrag kann klein oder sehr groß sein. Es können auch mehrere oder alle Parameter gleichzeitig verändert werden.

Die Veränderung von nur wenigen Parametern war in den Tests einfach zu langsam. Man findet zwar schnell ein Individuum, die etwas besser als ihr Original ist, jedoch ist der Unterschied sehr, sehr gering, so dass man genau testen muss, um ihn überhaupt korrekt festzustellen. Zudem benötigt man bereits sehr viele Evolutions-Schritte, um eine deutliche Verbesserung gegenüber dem Startwert zu messen.

Daher wird in der aktuellen Version jeder Parameter verändert, die Größe der Veränderung ist variabel. Wählt man eine große Veränderungs-Rate, so kommt es auf Grund der Zufälligkeit öfter vor, dass alle Mutationen schlechter sind als ihr Original. Jedoch besteht auch die Chance, dass schnell Parameter getroffen werden, die deutlich besser sind. Wählt man die Rate kleiner, so ist die Chance, eine Verbesserung zu erzielen, größer, jedoch sind die Verbesserungen nicht sehr groß. Es empfiehlt sich daher, zu Beginn des Algorithmus eine große Schrittweite zu wählen, und diese immer mehr zu verkleinern. Dadurch nähert man sich besser einem Optimum an – zunächst sehr schnell, aber ungenau, und dann immer feiner.

Die Veränderungs-Raten wurden nach eigenem Ermessen festgelegt. Zu Beginn des Algorithmus wird jeder Parameter (getrennt voneinander) bis zu einem Wert von 50 nach oben oder unten verändert. Nach einer bestimmten Anzahl von Generationen wird dieser Wert auf 30 und schließlich auf 10 reduziert.

Um zu verhindern, dass die nächste Generation schlechter ist als die aktuelle, da alle Mutationen ungünstig verlaufen, werden in jedem Spiel ein oder mehrere Originale hinzugefügt. Falls alle Mutationen schlecht sind, sollte ein Original-Spieler die beste Bewertung erhalten.

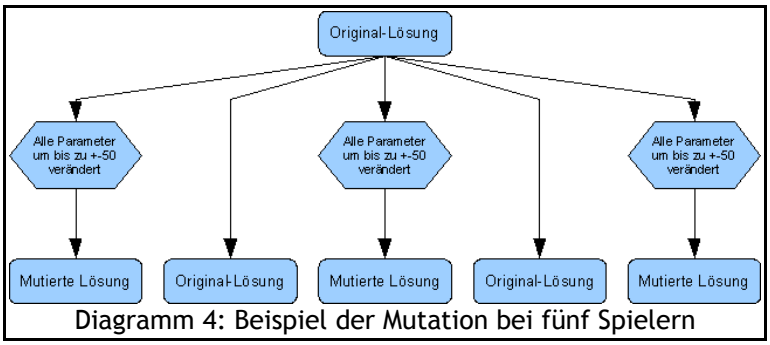

Diagramm 4: Beispiel der Mutation bei fünf Spielern

3.6 Rekombination

Die Rekombinations-Strategie bestimmt, wie aus mehreren vorhandenen Individuen neue Individuen generiert werden.

Da immer nur ein Individuum für die nächste Generation ausgewählt wird, und daher eine Generation nur aus einem Individuum besteht, kann keine Rekombination statt finden. Die Veränderung der Individuen funktioniert alleine über Mutationen.

3.7 Bewertung

Dieser Abschnitt beschreibt die Bewertung eines künstlichen Gegners, nicht die Bewertung eines Zuges.

Ein künstlicher Gegenspieler wird auf Basis seiner Leistungen im Spiel bewertet. Messbar sind dabei die erreichte Punktzahl und die Platzierung gegenüber der anderen Mitspieler. Bewertet wird nur die Platzierung.

Anfangs wurden nur die Siege der einzelnen Spieler gezählt. Dies führte jedoch dazu, dass ein Spieler, der in einem Spiel mit fünf Spielern stets den zweiten Platz belegte, schlechter bewertet wurde als ein Spieler, der einmal einen Sieg erringen konnte und ansonsten immer den letzten Platz belegt. Meiner Ansicht nach ist dies nicht korrekt, darum werden nun Punkte vergeben, je nach Platzierung.

Ich habe die Punkteverteilung relativ willkürlich gewählt. Wichtig war mir nur, dass ein Sieg und ein zweiter Platz sich deutlich von den anderen Platzierungen abheben, und dass zwei zweite Plätze mehr wert sind als ein Sieg und ein letzter Platz. Der Erstplatzierte bekommt 10 Punkte, der Zweite 6 Punkte, der Dritte 3, der Vierte 2 und der Letzte schließlich nur einen Punkt. Um nun den Glücksfaktor einzudämmen und eine sichere Auswertung zu gewährleisten werden immer sehr viele Spiele gespielt, bevor eine Auswahl bei den Spielern getroffen wird. Bei der alten Version waren über 10000 Spiele üblich; da die neue Version jedoch viel langsamer ist, wurde die Spielanzahl auf 1000 – 2000 begrenzt. Zusätzlich wird der Spieler, der als erstes an der Reihe ist, bei jedem Spiel gewechselt, so dass kein Spieler einen eventuellen Vorteil besitzt. Die Genauigkeit von diesem Verfahren wurde getestet, indem zwei gleiche Spieler gegeneinander gespielt haben, und schließlich die Ergebnisse verglichen und ein Quotient gebildet wurde. Da die beiden Spieler gleich stark spielen, sollte ein Wert nahe eins heraus kommen. Je größer die Schwankungen um diesen Wert sind, desto ungenauer ist die Messung. Gegebenenfalls kann man dann die Spielanzahl erhöhen. Bei 1000 Spielen bewegen sich die Werte aber schon sehr sicher zwischen 0.98 und 1.02, bei 2000 Spielen wird es noch ein wenig genauer, dort liegen die Werte zwischen 0.99 und 1.01. Diese Genauigkeit sollte ausreichen.

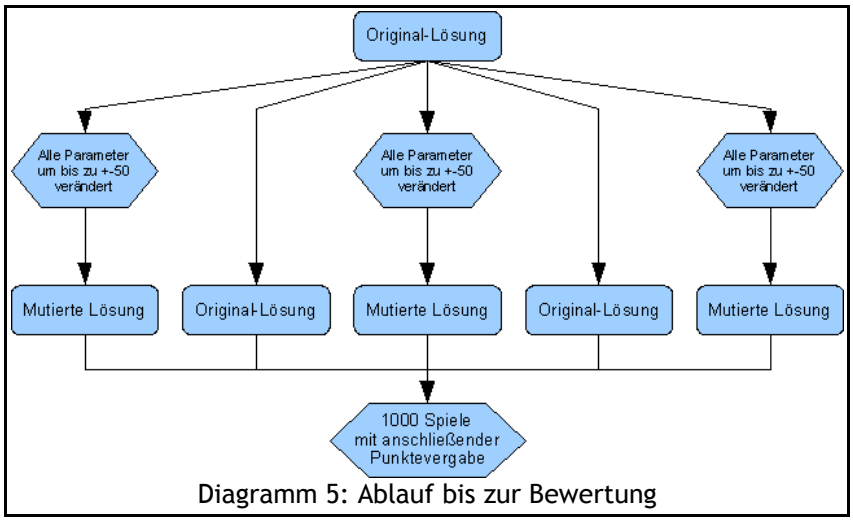

Diagramm 5: Ablauf bis zur Bewertung

3.8 Selektion

Die Selektions-Strategie legt fest, welche Individuen aus der gegebenen Auswahl an Mutationen bestimmt werden, um die nächste Generation zu formen. Diese Selektion findet nach der Bewertung aller Mutationen statt.

Bei dieser Aufgabenstellung ist keine komplizierte Selektion notwendig. Jede Generation besteht nur aus einem Individuum, außerdem besteht nach der Bewertung eine eindeutige Reihenfolge, die sich aus den vergebenen Punktzahlen ergibt. Daher wird generell nur das beste Individuum ausgewählt.

In dem Fall, dass nur zwei Spieler gegeneinander antreten, ist dies ohnehin die einzige Methode. Es würde keinen Sinn machen, noch das andere Individuum mit zu berücksichtigen. Falls jedoch mehrere Spieler beteiligt sind, wäre es als Beispiel denkbar, die zwei besten Individuen auszusuchen und ein Cross-Over durchzuführen, also beide Individuen geschickt zu einem neuen Individuum zu vereinen. Dies kann sehr gut funktionieren, ich halte das Vorgehen in diesem konkreten Fall allerdings nicht für sinnvoll. Wenn ein künstlicher Gegenspieler ein gutes Ergebnis erzielt, dann liegt dies an der günstigen Anordnung seiner Parameter. Da jedoch fast alle Parameter voneinander abhängig sind, sollte man diese nicht auftrennen und mit einem anderen Satz kombinieren. Als kleinen Test habe ich zwei unterschiedliche, gute Individuen, mehrmals zu einem Individuum kombiniert, indem ich die Parameter zufällig aus den Parametern der zwei Ausgangs-Individuen zusammengestellt habe. Bei den anschließenden Prüfungen waren die Ergebnisse jedoch alle sehr schlecht.

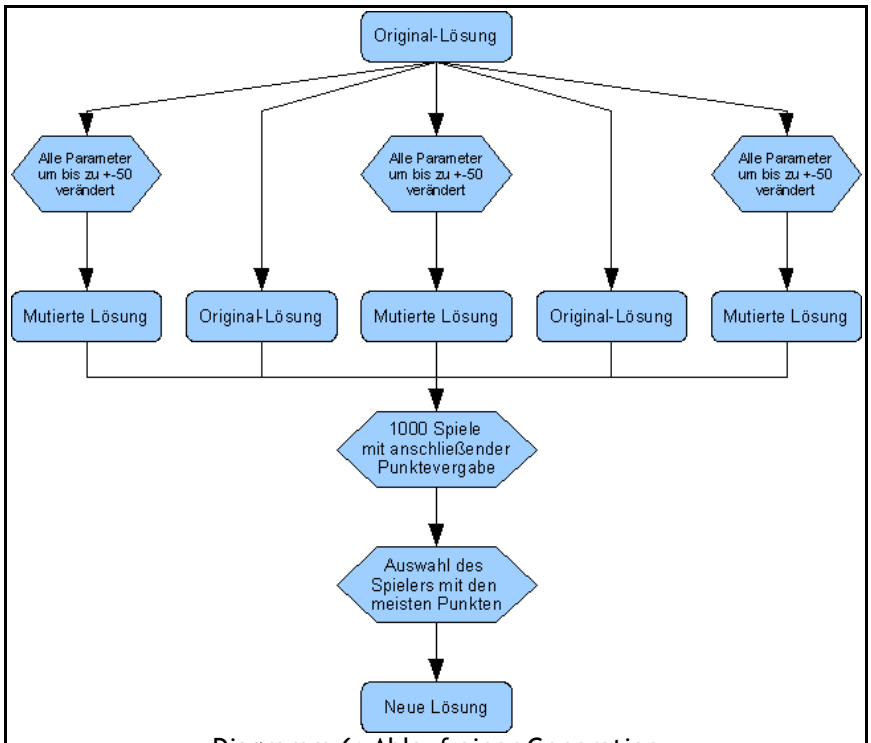

Diagramm 6: Ablauf einer Generation

4 Einschätzung der Spielstärke

4.1 Vorgehensweise

Es werden vier verschiedene Gegenspieler entwickelt: Einen für das Spiel mit zwei Spielern, einen für das Spiel mit drei Spielern, ebenso mit vier und fünf Spielern. Nach mehrmaligem Spielen des Spiels nahm ich an, dass die eigene Spielart je nach Spieleranzahl stark variiert, deshalb gibt es für jede Situation einen eigenen Gegenspieler.

Es stellt sich nun die Frage, wie schnell sich der künstliche Gegenspieler durch die Anwendung des evolutionären Algorithmus verbessert, und welche Spielstärke er letztendlich erreicht. Den ersten Teil dieser Frage kann man noch recht einfach beantworten; eine Aussage über die tatsächliche Spielstärke lässt sich nur sehr schwer treffen.

Um den Fortschritt des Gegenspielers zu kontrollieren, werden in regelmäßigen Abständen, beispielsweise nach jeweils 10 Generationen, Kontrollrunden durchgeführt. In diesen Kontrollrunden treten Spieler der aktuellen Generation zuerst gegen Spieler mit festgelegten Parametern an. Da jede Testreihe, wie schon erwähnt, immer für eine bestimmte Spieleranzahl durchgeführt wird, muss auch hier die korrekte Spieleranzahl eingehalten werden. Nach sehr vielen Spielen wird die durchschnittlich erreichte Punktzahl aller aktuellen Spieler mit der durchschnittlichen Punktzahl aller Vergleichs-Spieler verrechnet und ein Quotient gebildet. Je höher dieser Quotient ausfällt, desto mehr Spiele haben die aktuellen Spieler gegen die Vergleichs-Spieler gewonnen. Ein Wert unter eins hieße allerdings, dass sie schlechter abgeschnitten haben.

Zwar sind beim Spiel mit fünf Spielern Werte über 2 möglich, es ist jedoch nahezu unmöglich, dass sehr hohe oder niedrige Werte gemessen werden. Ein Wert um 1,4 – 1,5 ist bereits recht hoch und bedeutet, dass vom aktuellen Spieler 40 – 50 Prozent mehr Punkte gewonnen wurden.

Nachdem der Test gegen die festgelegten Vergleichs-Spieler abgeschlossen ist, wird der erreichte Wert notiert. Normalerweise sollte der Wert höher sein als der Wert in der letzten Kontrollrunde. Je höher die Werte allerdings werden, umso öfter kommt es vor, dass ein eigentlich verbesserter Gegenspieler keine höhere Punktzahl gegen den Testspieler erreicht, als ein schlechterer Gegenspieler. Da das Kartenspiel immer noch zum Teil vom Glück abhängt, ist es nun mal nahezu unmöglich, wirklich alle Spiele zu gewinnen. Und auch wenn Spieler a klar besser ist als Spieler b, ist es möglich, dass beide gegen Spieler c gleich gut abschneiden.

Es wird deshalb noch ein weiterer Test in jeder Kontrollrunde durchgeführt, und zwar wird der aktuelle Spieler mit dem Spieler der letzten Kontrollrunde verglichen. Dadurch wird direkt geprüft, wie viel sich der Spieler seit der letzten Kontrolle verbessert hat. Hier sollte zumindest ein Wert über eins heraus kommen, falls seit der letzten Kontrolle ein paar verbessernde Mutationen entstanden sind. Und je länger die Pausen zwischen den Kontrollrunden sind, umso höher sollte der Wert liegen.

Für den abschließenden Test wurden für jede Spieleranzahl zuerst drei Testreihen durchgeführt. Jede Testreihe bestand aus vielen Generationen und Kontrollrunden, wobei in jeder Generation 2000 Spiele und in jeder Kontrollrunde insgesamt 8000 Spiele durchgeführt wurden.

Es wurden für jede Reihe zuerst 100 Generationen berechnet, nach jeweils 10 Generationen folgte eine Kontrollrunde. Bei Bedarf wurde die Testreihe nun erweitert oder unter geänderten Bedingungen neu gestartet, falls noch kein ausreichendes Ergebnis abzulesen war.

Um die absolute Spielstärke des Gegenspielers zu bestimmen, fehlen leider die Möglichkeiten des Vergleichs. Es gibt zur Entstehungszeit dieser Arbeit noch keine anderen Programme für das Spiel „Chinesische Mauer“, in denen ein künstlicher Gegner vorhanden wäre. Ansonsten könnte man die beiden Programme gegeneinander antreten lassen und hätte auf diese Weise einen Vergleich. Ein Vergleich mit menschlichen Spielern ist sehr ungenau, noch dazu gibt es keine bekannten guten Spieler dieses Spiels, die eine professionelle Aussage über die Spielstärke treffen könnten.

So bleibt lediglich die Möglichkeit, Tests mit beliebigen menschlichen Spielern durchzuführen. Falls der künstliche Spieler bei diesen Tests sehr oft sehr schlecht abschneidet, ist dies schon eine aussagekräftige Feststellung, denn dies bedeutet in der Regel, dass der Gegenspieler schwach ist. Falls jedoch ein ausgeglichenes oder positives Ergebnis für den Gegenspieler entsteht, so ist damit nicht viel bewiesen.

4.2 Ergebnisse

Nach dem Durchlaufen mehrerer umfangreicher Testreihen muss ich feststellen, dass lediglich die Ergebnisse bei zwei Spielern einen Erfolg vermuten lassen.

4.2.1 Das Spiel mit zwei Spielern

Obwohl die Mutationen zufällig verlaufen und beim Spiel mit zwei Spielern immer nur ein einziges mutiertes Individuum mit dem Original-Individuum verglichen wird, haben bei allen drei Tests 100 Generationen völlig ausgereicht, um eine Leistungs-Steigerung von etwa 40 Prozent gegenüber dem Vergleichs-Spieler zu erreichen.

```
Beispiel-Auszug aus Konsolen-Ausgabe:

Test Beginn, 2 Spieler, 100 Generationen, je 10 pro Kontrollrunde

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 0.9980206913397851
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 0.6925903194992518

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.1682074410895504
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 1.1707797964016917

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.2624658372984778
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 1.0820822961963348

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.2680804205740266
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 1.0156041691916131

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.2614639182677003
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 1.006905690937201

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.2412401009631884
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 0.9949310015755329

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.311065425758148
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 1.057299983323258

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.4060206105821944
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 1.126743854106039

Test Ende
```

Als diese Grenze erreicht wurde, konnten von Kontrollrunde zu Kontrollrunde (also jeweils 10 Generationen) keine echten Fortschritte mehr gemessen werden.

```
Beispiel-Auszug aus Konsolen-Ausgabe:

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.4008226879249794
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 0.9923696847486014

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.4161061413000506
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 1.0003732458454944

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.4271349989636315
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 0.9861365948813043

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.4190677896645685
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 1.0163510343750053
```

Das folgende Diagramm zeigt noch einmal anschaulicher die Entwicklung der Spielstärke dieses Gegenspielers im Laufe der Generationen.

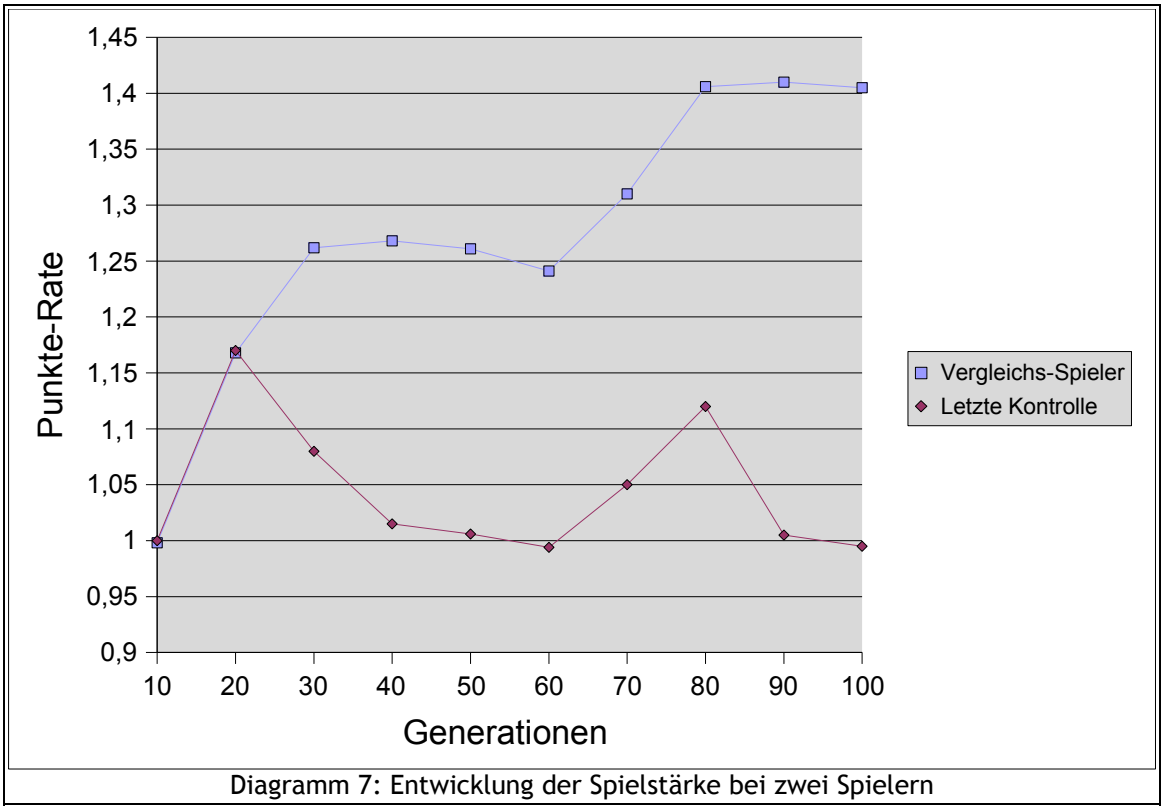

Diagramm 7: Entwicklung der Spielstärke bei zwei Spielern

Um zu überprüfen, ob doch noch Fortschritt statt findet, wurden von einer Testreihe weitere 300 Generationen berechnet, also insgesamt 400. Das folgende Diagramm zeigt nochmals, dass in diesen zusätzlichen Generationen kein Fortschritt gemessen werden konnte.

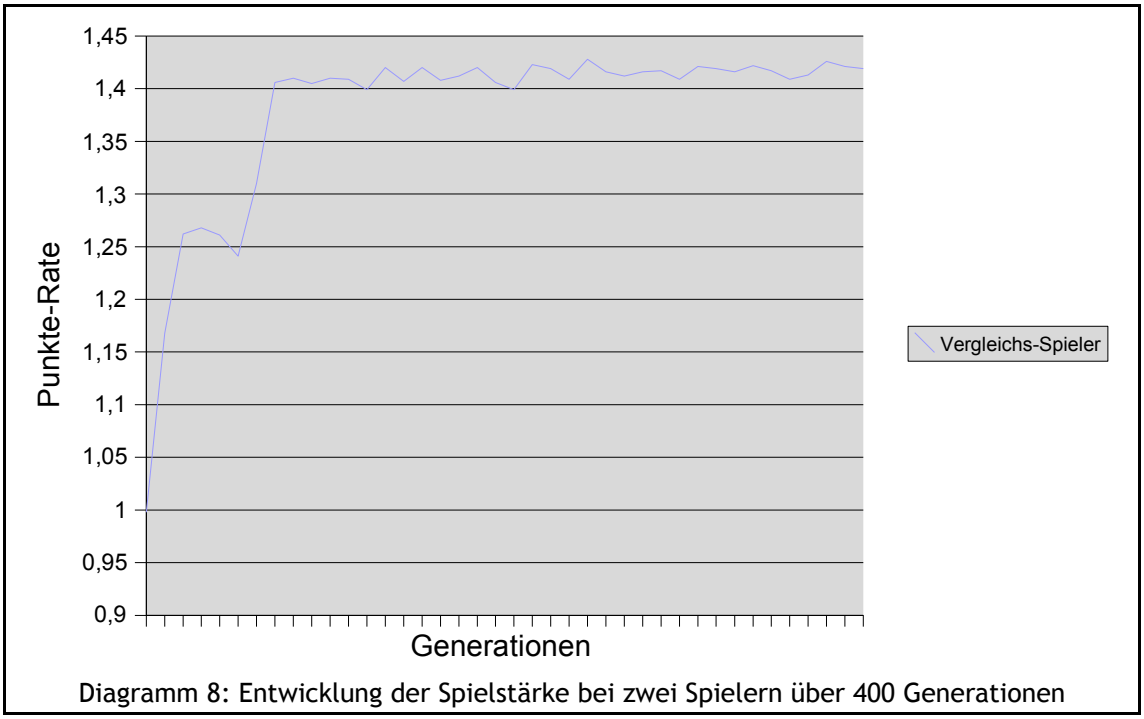

Diagramm 8: Entwicklung der Spielstärke bei zwei Spielern über 400 Generationen

Danach wurde der resultierende Gegner mit den drei Gegnern, die aus jeweils 100 Generationen entstanden sind, verglichen. Dadurch wurde aus diesen vier Spielern der Beste ermittelt.

Der Gegenspieler, der 400 Generationen durchlief, konnte alle anderen Spieler besiegen. Dieses Ergebnis kam nur deshalb überraschend, da seine Wertung gegen den festen Vergleichs-Spieler sogar geringfügig schlechter ausfiel als die Wertung zweier seiner Gegner. Dies zeigt, dass diese Wertung keine Auskunft über die tatsächliche Spielstärke gibt, sondern höchstens eine Tendenz aufzeigen kann, denn der Gegenspieler hat diese Wertung in den zusätzlichen 300 Generationen nicht verbessert, und trotzdem hat er seine Gegner besiegen können.

Bei der Gelegenheit wurde noch überprüft, ob die Annahme richtig war, dass für jede Anzahl an Spielern ein eigener Gegenspieler entwickelt werden muss. Ein Spieler, der 100 Generationen mit fünf Spielern durchlaufen hat, wurde gegen den Spieler mit 400 Generationen aus den Tests mit zwei Spielern getestet. Die Annahme wurde bestätigt:

Beim Test in einem Spiel mit fünf Mitspielern hat der erste Spieler klar dominiert und konnte eine sehr hohe Wertung von 1,6 gegenüber dem zweiten Spieler erzielen. Beim Test mit zwei Spielern jedoch lief es umgekehrt – der erste Spieler verlor die meisten Spiele und erzielte eine niedrige Wertung von 0,78.

Die absolute Spielstärke des zwei-Spieler Gegenspielers ist immer noch unklar. Ich habe zum Abschluss der Arbeit zehn Spiele gegen den Gegenspieler gespielt und konnte davon drei für mich entscheiden, ein Spiel ging unentschieden aus. Das ist ein einigermaßen gutes Ergebnis, denn ich bin aufgrund der langen Zeit, in der ich mich mit dem Spiel auseinander gesetzt habe, zumindest kein Anfänger mehr. Trotzdem habe ich eigentlich gehofft, dass der Gegenspieler ein deutlicheres Ergebnis erzielt. Aber natürlich muss auch angemerkt werden, dass diese zehn Spiele nicht umfassend repräsentativ sind. Um eine klarere Aussage zu treffen müsste man längerfristige Tests mit vielen menschlichen Spielern und über viele hundert Spiele durchführen, doch dazu fehlte die Zeit.

4.2.2 Das Spiel mit mehreren Spielern

Auffällig war, dass beim Spiel mit ungerader Spieleranzahl die Verteilung der Spieler bei den Kontrollrunden einen überraschenden Unterschied machte: Spielten zum Beispiel zwei aktuelle Spieler gegen einen Vergleichs-Gegner, so schnitten sie immer deutlich schlechter ab, als im darauf folgenden Test, bei dem die Verteilung genau anders herum war. Da bei jeder Kontrolle immer beide Tests durchgeführt werden und der Durchschnitt der Ergebnisse gebildet wird, sollte dies jedoch keinen besonderen Einfluss haben.

Beim Spiel mit drei oder mehr Spielern ließen die gemessenen Werte vermuten, dass der Algorithmus kein gutes Ergebnis hervorbringt. Es ist auch im frühen Stadium

des Algorithmus häufig vorgekommen, dass ein Gegenspieler beim Vergleich mit dem Gegenspieler 10 Generationen vorher ein negatives Ergebnis aufwies, und auch die Messung gegen den Vergleichs-Spieler brachte einen schlechteren Wert als vorher. Dabei gab es zwischen den Verläufen beim Spiel von drei, vier und fünf Spielern keine erkennbaren Unterschiede.

```
Beispiel-Auszug aus Konsolen-Ausgabe:

Test Beginn, 5 Spieler, 50 Generationen, je 10 pro Kontrollrunde

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.105560988810864
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 1.0278427657274802

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.0141532126018384
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 0.9918342595748582

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.0526462264401104
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 1.034297846622654

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.0242913083601761
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 1.0186267464092285

Schnitt Verbesserungsfaktor gegen festgelegte Testwerte: 1.0365951185930218
Schnitt Verbesserungsfaktor gegen vorherige Evolution: 0.9719814917084553
```

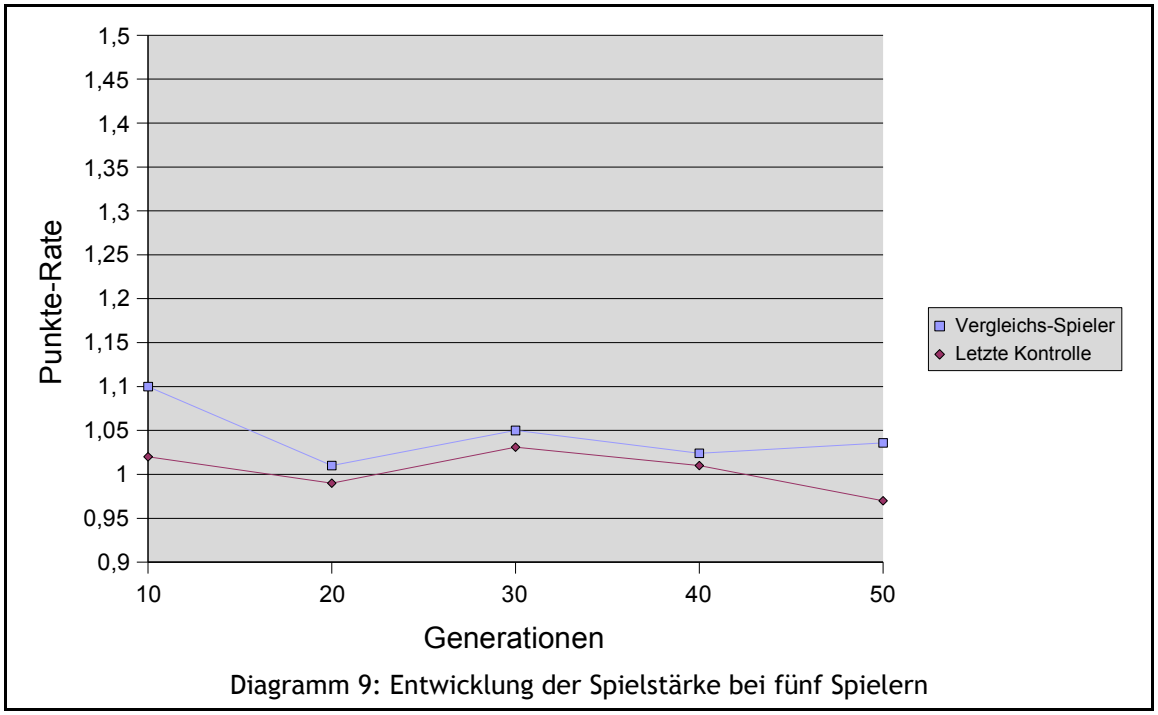

Diagramm 9: Entwicklung der Spielstärke bei fünf Spielern

Die erste Vermutung war, dass durch einen eventuell erhöhten Glücksfaktor einfach deutlich mehr Generationen benötigt werden, um den gewünschten Erfolg zu erzielen. Doch auch Tests mit deutlich höherem Zeitaufwand zeigten keine guten Ergebnisse. Das folgende Diagramm 10 zeigt die Ergebnisse eines Testlaufs mit drei Spielern, der auf einem 2,08 Ghz-Rechner fast 6 Stunden Zeit in Anspruch nahm.

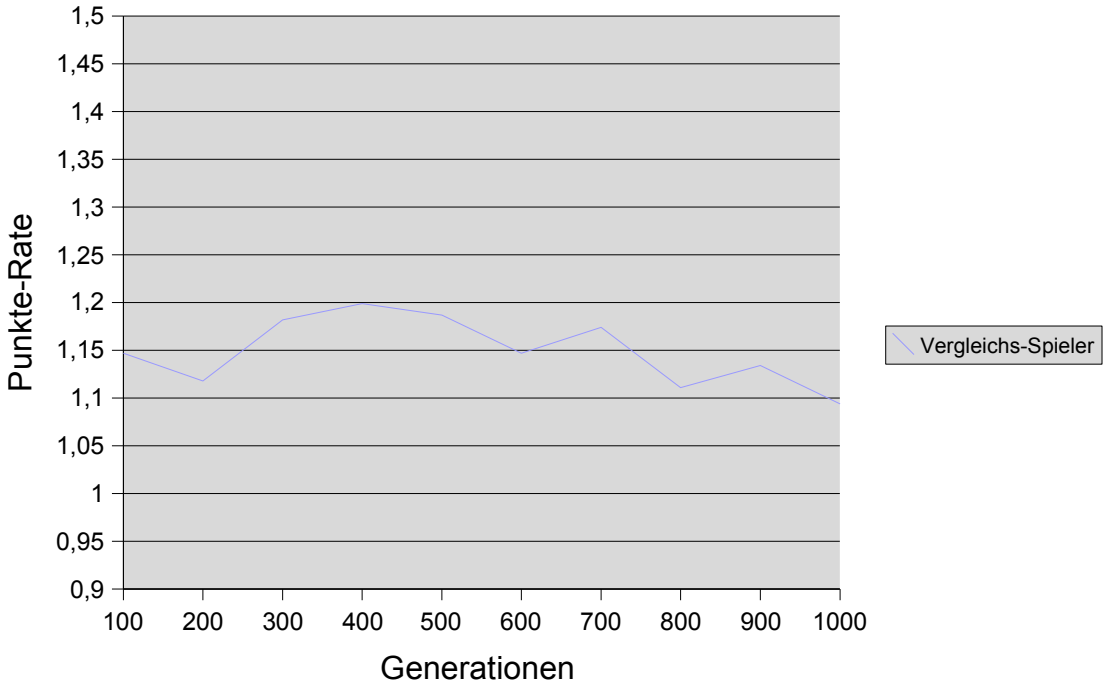

Diagramm 10: Entwicklung der Spielstärke bei drei Spielern über langen Zeitraum

Es wurden ähnliche Tests mit vier und fünf Mitspielern durchgeführt, die jeweils noch viel mehr Zeit in Anspruch nahmen, doch vergleichbare Ergebnisse zeigten.

Um auszuschließen, dass alle möglichen Initialwerte in einer Sackgasse enden, aus der nicht mehr entkommen werden konnte, wurden die Streuungen der Parameter beim Start und die Streuung der Parameter bei einer Mutation stark erhöht, doch auch diese Maßnahmen brachten wenig Erfolg. Sie führten nur dazu, dass manchmal der Initialwert so schlecht war, dass der Gegenspieler kaum noch vernünftige Züge durchgeführt hat. Doch nach wenigen Generationen wurde wieder ein Zustand angenommen, der kaum noch Verbesserung brachte.

Zu guter Letzt habe ich wieder selbst einige Spiele gegen die entwickelten Gegenspieler gespielt, um nachzuprüfen, ob vielleicht doch ein guter Gegner heraus gekommen ist, und die Messungen dies nur nicht erfassen konnten.

Gegen alle Arten von Gegenspielern erzielte ich mittelmäßige bis gute Ergebnisse, das bedeutet, dass ich von zehn Spielen im Schnitt etwa vier bis fünf Spiele gewinnen konnte, und in den restlichen Spielen zumindest eine gute Platzierung erreichte. Auch hier fehlen ausführlichere Tests gegen vergleichbare, andere Gegenspieler oder menschliche Spieler, um ein aussagekräftiges Ergebnis zu erzielen.

5 Diskussion

5.1 Einschätzung des Ergebnisses

Das Ergebnis dieser Arbeit fasse ich noch einmal kurz zusammen:

- Die Messungen zeigten bei zwei Spielern stets Erfolg, und zwar schon nach relativ wenigen Generationen
- Nach noch mehr Generationen waren die Testergebnisse zwar beinahe gleich, jedoch zeigte der direkte Vergleich doch eine Leistungssteigerung
- Bei mehr als zwei Spielern ergaben die Messungen, dass der Gegenspieler sich nicht, wie zuvor erwartet, stets verbessert, sondern eher zufällig etwas besser oder etwas schlechter wurde, doch insgesamt auf einem durchschnittlichen Niveau stagnierte
- Auch nach sehr vielen Generationen stellte sich keine messbare Leistungssteigerung ein
- Beim manuellen Spiel mit zwei Spielern (also gegen einen computergesteuerten Spieler) konnte der Gegner leicht mehr Spiele gewinnen als ich
- Gegen drei oder mehr Gegner konnte ich jeweils einige Siege verbuchen, und mich ansonsten auf einem guten Platz halten

Die Fragen, die sich nun stellen:

- War der Algorithmus bei zwei Mitspielern tatsächlich erfolgreich?
- War der Algorithmus bei mehreren Mitspielern tatsächlich NICHT erfolgreich?
- Wenn ja: Warum nicht, und warum gab diese Unterschiede?

Die Fragen werden nun einzeln betrachtet:

5.1.1 War der Algorithmus bei zwei Mitspielern tatsächlich erfolgreich?

Die Messungen ergaben eine stetige Verbesserung, und persönlich war ebenfalls eine Verbesserung spürbar. Dass die Messergebnisse kein Zufall waren, wurde ausgeschlossen, indem der Test mehrmals wiederholt wurde, und jedes Mal vergleichbare Ergebnisse zu beobachten waren. Lediglich die Qualität war leicht unterschiedlich, doch dies ist ganz normal, da durch die zufälligen Mutationen die Geschwindigkeit der Verbesserungen nicht immer gleich ist.

Also lässt sich ein Fazit ziehen: Der Algorithmus war erfolgreich. Er hat tatsächlich immer eine Verbesserung gebracht, und je länger der Algorithmus lief, desto besser war der resultierende Spieler. Dass der finale Gegenspieler trotzdem keine überragende Spielstärke erreicht hat, liegt augenscheinlich an meiner Implementierung. Der Gegenspieler kann ja höchstens so gut werden, wie meine Aufteilung der Ge-

wichte und meine internen Algorithmen es zulassen. Diese sind nicht optimal und geben daher noch großen Spielraum für Verbesserungen.

5.1.2 War der Algorithmus bei mehreren Mitspielern tatsächlich NICHT erfolgreich?

Die Bewertungs-Funktion wurde sehr oft geändert und überarbeitet, um einen möglichst fairen Vergleich beim Spiel mit mehreren Spielern zu ermöglichen, und trotzdem konnte keine stetige gemessen werden. Meiner Meinung nach ist aber gerade der Vergleich mit dem festgelegten Testspieler tatsächlich fair und aussagekräftig, und die Messungen sind auch allesamt reproduzierbar, somit nicht zufällig. Die Antwort muss also lauten: So ist es – der Algorithmus hat nicht den gewünschten Effekt gebracht. Es kam zwar immer ein einigermaßen passabler Gegenspieler heraus, doch der erwünschte Fortschritt konnte nicht festgestellt werden.

5.1.3 Wenn ja: Warum nicht, und warum gab diese Unterschiede?

Was ist beim Spiel von zwei Spielern so anders als bei Spiel von drei oder mehr Spielern, dass solch ein Unterschied zu Stande kommen kann? Ich werde im Folgenden einige Ideen aufzählen und auf sie eingehen:

- Der evolutionäre Algorithmus funktioniert bei zwei Spielern besser
- Die Routinen im Programm funktionieren bei zwei Spielern besser
- Das Spiel „Chinesische Mauer“ ist bei mehr als zwei Spielern sehr zufällig

Der Grundgedanke des evolutionären Algorithmus eignet sich für jede Spieleranzahl gleich, es ist grundsätzlich egal, ob aus zwei Spielern des beste Spieler ausgesucht wird oder aus fünf. Die Vorgehensweisen der Gegenspieler im Programm sollten auch keine entscheidende Rolle spielen, da die Regeln im Spiel bei jeder Spieleranzahl gleich ist. Dass die Spielweise natürlich variiert ist klar, doch dies wird durch die variablen Parameter, die sowohl eine aggressive als auch eine defensive Spielweise ermöglichen, abgedeckt. Dass dies funktioniert haben die Tests mit den Gegenspielern aus verschiedenen Testreihen mit unterschiedlicher Spieleranzahl gezeigt. Das Spiel Chinesische Mauer scheint jedoch nach eigenem Gefühl tatsächlich mit mehr als zwei Spielern zufälliger zu sein. Ich denke, dass ein klar besserer Spieler sich auch in einem fünf Spieler-Spiel gegen seine Gegner durchsetzen kann, so dass er sich zumindest allgemein besser platzieren kann. Falls durch eine Mutation ein solch klar besser Spieler entsteht, so wird er auch bestimmt ausgewählt, und es ist ein Fortschritt zu verzeichnen. Falls jedoch alle Spieler eher gleich stark sind und es keine eindeutigen Unterschiede gibt, so ist es nach meinem Erachten sehr zufällig und von sehr vielen unbeeinflussbaren Faktoren abhängig, wer die meisten Punkte bekommt. Je nachdem, wie die anderen Mitspieler sich verhalten, wird das eigene

Spiel einfach zu sehr beeinflusst, darum sind leicht bessere Spieler kaum von den anderen zu unterscheiden. Es passiert deshalb zu leicht, dass ein eigentlich leicht schlechterer Spieler gewinnt.

Noch dazu kommt, dass bei mehr als zwei Spielern die Methode, wie man die resultierenden Gegenspieler kontrolliert, zu weiteren Faktoren führt, die man schlecht einschätzen kann. Wenn man einen resultierenden Spieler, der auf fünf-Spieler-Spiele trainiert war, mit dem Vergleichs-Spieler messen möchte, kann man ja nicht einfach die Beiden in einem zwei-Spieler-Spiel gegeneinander antreten lassen. Es muss auf jeden Fall ein Spiel mit fünf Spielern sein, doch die Messungen haben gezeigt, dass schon alleine die Anzahl der jeweiligen Spieler das Ergebnis verändern, auch wenn man die Ergebnisse der Spielergruppen jeweils mittelt. In dem Fall, der am Anfang des Kapitels 4.2.2 erwähnt wird, spielt ein Gegenspieler deutlich schlechter, wenn noch ein weiterer Vertreter von ihm an der Partie teilnimmt. Sind außer ihm nur Vergleichs-Spieler anwesend, gewinnt er viel mehr Partien. Es ist also offensichtlich, dass selbst die Zusammenstellung der anderen Spieler das Ergebnis eines einzelnen Spielers maßgeblich beeinflussen können. Wie soll man in einem solchen Fall bei fünf Spielern wirklich sicher sein, dass der Spieler mit den meisten Punkten tatsächlich der beste Spieler ist? Es könnte durchaus sein, dass dieser Spieler bei einer anderen Konstellation der selben Gegner schon wieder schlechter abschneidet.

Bei einem weiteren Test kam auch genau dieser Umstand zu Tage: Bei einem Spiel mit fünf Spielern gewinnt ein Spieler mit veränderten Parametern recht deutlich. Mit ihm nehmen an der Partie noch zwei beliebige Gegner und zwei Vergleichs-Gegner teil. Man sollte annehmen, dass dieser Spieler seine Gegner jeweils besiegen kann.

Im Vergleichs-Spiel, bei dem nur Vertreter von diesem Gegenspieler und Vergleichs-Gegner in verschiedenen Konstellationen aufeinander treffen, gewinnen die Vergleichs-Spieler im Schnitt über zehn Prozent mehr Punkte.

Keine der drei getroffenen Aussagen am Anfang dieses Abschnitts ist die Antwort auf die Frage, warum das Ergebnis nicht zufriedenstellend ausfällt – und doch ergeben sie zusammen den Grund. Damit der evolutionäre Algorithmus funktionieren kann, muss man möglichst präzise aussagen können, welches Individuum besser oder schlechter ist. Das funktioniert bei diesem Spiel bei mehreren Gegnern aber nicht. Grund hierfür ist der zu hohe Einfluss der anderen Mitspieler, der zu oft verfälscht, welcher Spieler wie gut einzuschätzen ist. Und das liegt zum Teil am Spiel selbst, das mit mehreren Spielern tatsächlich viel von den Aktionen der Gegner abhängig ist, zum anderen kann auch die einprogrammierte Vorgehensweise der Gegenspieler dafür sorgen, da die nicht ausgereift ist und an einigen Stellen Schwachstellen aufweist.

5.2 Verbesserungsmöglichkeiten

5.2.1 Routinen der künstlichen Gegenspieler überarbeiten

Die fest einprogrammierten Teile des künstlichen Gegenspielers können noch einmal überdacht und verändert werden.

Fest einprogrammiert ist die Verwendung der Drachen-Karte. Es wird nicht jeder mögliche Zug mit der Drachen-Karte probiert, sondern für jede Reihe wird mittels fest geschriebener Prozedur die günstigste Platzierung ermitteln und nur dieser Zug benutzt. Das Durchprobieren jeder möglicher Position könnte natürlich noch eine bessere Verwendung als die von der Prozedur errechneten Position finden, dafür wäre der Rechenaufwand deutlich höher, so lange sich eine Drachen-Karte in der Hand befindet. Falls die Position, die von der Prozedur bestimmt wird, aber wirklich einen schlechten Zug ergibt, dann wird der Zug auch nicht benutzt, da die Bewertung dieses Zuges schlecht wäre. Trotzdem könnte eine Änderung an diesem Teil des Programms noch eine Verbesserung bringen.

Auch fest einprogrammiert ist der Algorithmus zur Entscheidung, welches Märkchen genommen wird in dem Fall, dass bei einem Sieg in einer Reihe noch zwei Märkchen zur Auswahl stehen. Zur Zeit gibt es dort eine einfache Abfrage die prüft, ob man beim Nehmen des schwächeren Märkchens noch Chancen auf das zweite Märkchen besitzt. Ist dies nicht der Fall, wird gleich das hohe Märkchen genommen. Diese Vorgehensweise erscheint mir recht einleuchtend und intuitiv, dennoch könnte sie noch verfeinert werden.

Der Hauptansatzpunkt für Verbesserungen sind trotz allem die verschiedenen Gewichte. Die Wahl der Merkmale, die dann von den Parametern gewichtet werden, sind sehr entscheidend für die resultierende Spielstärke, und genau diese Wahl ist alles andere als einfach. Bei meinem Programm ist die Anzahl der Parameter sehr gering. Ich habe sie nach einigen Spielen gegen mich selbst auf der Basis meiner eigenen Überlegungen gewählt, und sie ermöglichen meiner Meinung nach auch die wichtigsten Taktiken, die mir aufgefallen sind. Es ist zum Beispiel möglich, dass ein Gegenspieler sehr oft Karten nachzieht und damit eher passiv spielt – auch Spieler, die sehr oft oder bzw. sehr selten mehrere Karten zusammen auslegen könne entstehen. Nicht beachtet werden beispielsweise die Kartenanzahl der Gegner, oder welche Karten welcher Gegner bereits gelegt hat. Natürlich könnten auch diese Aspekte einen weiteren Vorteil bringen, dafür hat jedoch letztendlich die Zeit gefehlt. Diese wurde immer mehr für die teilweise sehr langen Tests, die ständig durchgeführt werden mussten, gebraucht.

5.2.2 Versuchsreihe mit dem Gegenspieler bei zwei Spielern

Um den Erfolg und die tatsächliche Spielstärke des Gegenspielers bei zwei Spielern festzustellen, wäre eine umfangreichere Testreihe mit vielen Spielern hilfreich. Eine Idee wäre eine Freeware-Version des Spiels, die die Ergebnisse der menschlichen Spieler über das Internet zusammenträgt, und so ermittelt, wer wie oft bereits das Spiel gespielt und dabei verloren oder gewonnen hat.

5.2.3 Verbesserungen am Programm selbst

Nicht wichtig für das Ergebnis des Algorithmus, aber dennoch hier kurz erwähnt seien ein paar Verbesserungsmöglichkeiten meines Programms, mit dem man das Spiel „Chinesische Mauer“ spielen kann. Es sind Verbesserungen zur Benutzerfreundlichkeit oder ansprechenden Gestaltung des Spiels, doch auf diese Aspekte habe ich während des Projekts keinen Wert gelegt, und so fanden sie keinen Weg in das fertige Programm.

Ein Tutorial, welches den Spieleinstieg erleichtert und dem Benutzer die Regeln anschaulich erläutert, fehlt ebenso wie Musik- und Sound-Effekte. Auch eine Option zum Abspeichern und Fortsetzen des Spiels wäre sehr schön. Etwas umfangreicher wäre die Programmierung der Möglichkeit, mehrere Spieler über das Internet gegeneinander antreten lassen zu können.

5.3 Andere Ansätze

Viele Spiele greifen gerade zu Beginn der Partie auf eine Datenbank von gespeicherten Spielen zurück, mit der sie die Eröffnungs-Konstellationen analysieren und auf Grund von statistischen Vergleichen ihren Zug wählen. Das bekannteste Beispiel wäre Schach, bei der bei vielen Eröffnungen umfangreiche Analysen über mögliche Gegenzüge und ihre weitere Entwicklung auf dem Spielverlauf existieren. Mit einer solchen Datenbank braucht ein Schachprogramm diese Analysen nicht selbst durchzuführen. Ähnliches gilt auch für Dame oder Go. Bei „Chinesische Mauer“ kann man solche Eröffnungen nicht anwenden, da es sehr viele verschiedene Verteilungen der Ruhmesmärkchen bei Spielbeginn gibt. Außerdem bekommt man immer andere Karten auf die Hand. Statistische Ansätze sind also nicht zu gebrauchen.

Oft wird auch ein Zug bewertet, indem die Möglichkeiten, wie der Gegner auf den eigenen Zug antworten könnte, in Betracht gezogen werden. Da jedoch, gerade zu Beginn der Partie, lediglich die Anzahl der Karten, die der Gegner auf der Hand hält, aber nicht die Art der Karten bekannt ist, sind solche Ansätze ebenfalls schwierig anzuwenden. Man müsste für zufällige Fälle der Kartenverteilung die Antwortmöglichkeiten des Gegners berechnen. Da der Gegner jedoch in jeder Runde zwei Aktionen hat, sind solche Berechnungen sehr umfangreich. In der Endphase des Spiels, wenn die Auswahl der Karten, die der Gegner noch in der Hand und im Nachziehstapel hat, sehr gering und überschaubar sind, wäre ein solcher Ansatz möglicherweise anzuwenden.

6 Literatur

[1] Wikipedia (deutsch):
http://de.wikipedia.org/wiki/Evolution%C3%A4rer_Algorithmus

[2] Fachhochschule Südwestfalen:
http://www.fh-meschede.de/public/willms/ea/index.html

[3] Wikipedia (englisch):
http://en.wikipedia.org/wiki/Evolutionary_algorithm

[4] http://www.its.leeds.ac.uk/projects/smartest/deliv3.html

[5] http://www.it.swin.edu.au/centres/ciscp/eas.php

[6] Boardgamegeek, eine Webseite für Gesellschaftsspiele:
http://www.boardgamegeek.com/game/22198

[7] Deutsches Regelwerk von der Webseite des Vertreibers KOSMOS

http://www.kosmos.de/kosmos/wrs/wrs.nsf/$WebFirstSource/FS764E6C936432D2A3C1257275002EF7B8/$File/Spielregel Chinesische Mauer.pdf

[8] Pollack/Blair: Co-Evolution in the Successful Learning of Backgammon Strategy
http://demo.cs.brandeis.edu/papers/bkg_ml.pdf